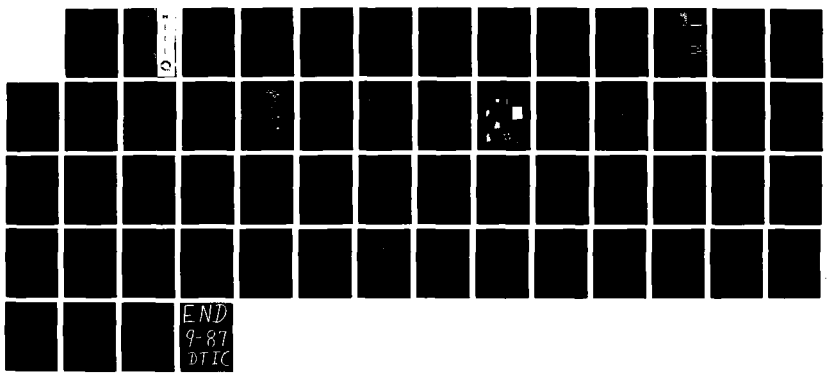
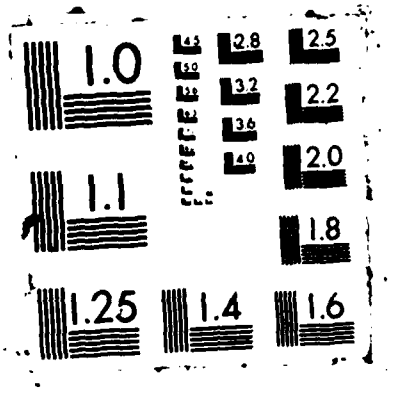
AD-A183 493 DEVELOPMENT OF COMPUTER VISION TECHNIQUES FOR AUTOMATIC FEATURE EXTRACTION(U) AUTOMETRIC INC FALLS CHURCH VA D K GORDON ET AL. 30 JAN 87 ETL-0451 1/1

UNCLASSIFIED F/G 12/9 NL




1.0
1.1
1.25
1.4
1.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.8

ETL-0451

Development of computer vision techniques for automatic feature extraction

Daniel K. Gordon
Richard F. Pascucci

January 1987

Prepared for
U.S. ARMY CORPS OF ENGINEERS
ENGINEER TOPOGRAPHIC LABORATORIES
FORT BELVOIR, VIRGINIA 22060-5546

E T L

U.S. ARMY ENGINEER TOPOGRAPHIC LABORATORIES
ETL

Destroy this report when no longer needed.
Do not return it to the originator.

The findings in this report are not to be construe s an official Department of the Army position unless so designat d by other authorized documents.

The citation in this report of trade names of commercially available products does not constitute official endorsement or approval of the use of such products.

Final Report

DEVELOPMENT OF COMPUTER VISION TECHNIQUES FOR AUTOMATIC FEATURE EXTRACTION

Prepared for:

U.S. Army Engineer Topographic Laboratories
Fort Belvoir, VA 22060

Prepared by:

AUTOMETRIC, INCORPORATED
5205 Leesburg Pike
Suite 1308/Skyline One
Falls Church, VA 22041

30 January 1987

SECURITY CLASSIFICATION OF THIS PAGE

ADA183493

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188
Exp Date Jun 30 1986

1a REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS N/A
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution is Unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) None.	5. MONITORING ORGANIZATION REPORT NUMBER(S) ETL-0451
6a. NAME OF PERFORMING ORGANIZATION Autometric, Incorporated 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION U.S. Army Engineer Topographic Laboratories
6c. ADDRESS (City, State, and ZIP Code) 5205 Leesburg Pike, Skyline One Falls Church, VA 22041	7b. ADDRESS (City, State, and ZIP Code) Fort Belvoir, VA 22060
8a. NAME OF FUNDING/SPONSORING ORGANIZATION --- 8b OFFICE SYMBOL (If applicable) ---	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER 225.3
8c. ADDRESS (City, State, and ZIP Code) ---	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO: --- PROJECT NO: --- TASK NO: --- WORK UNIT ACCESSION NO: ---

11 TITLE (Include Security Classification)
Development of Computer Vision Techniques for Automatic Feature Extraction (unclassified)

12 PERSONAL AUTHOR(S)
Daniel K. Gordon and Richard F. Pascucci

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Final Report	FROM 1 Feb 86 TO 15Jan 87	1987, January 30	54

16 SUPPLEMENTARY NOTATION
None.

17 COSATI CODES		
FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
- Computer Vision - SAR Imagery - Descriptor Sets
- Automatic Feature Extraction - Expert System

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

In previous work, 52 descriptors (feature identifiers) and 501 descriptor sets were identified as being used by image analysts for the characterization of features found in radar imagery. In the research investigation described herein, the descriptor sets were tested and validated. Following this, computer vision techniques were identified and developed to automatically recognize these descriptor sets. The identification procedure includes image preprocessing (e.g., edge enhancement, density slicing, neighborhood encoding and thinning), raster to vector conversion, and the processing of the resultant vector data (e.g., identification of points, lines, and areas, referred to as primitives, including a description of primitive size, shape, position and orientation). Finally, the relative positions of primitives were examined, and the similarity between groups of primitives and descriptor sets was quantified. The images used were softcopy versions of graphic, line-drawn examples of the descriptors that were identified in the previous work. The computer vision techniques that were developed have been demonstrated successfully, (cont'd on back)

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL E. James Books	22b TELEPHONE (Include Area Code) (202) 355-3039 22c OFFICE SYMBOL ETL-IM-L

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

19. (cont'd)

in a tightly controlled environment, on images containing little extraneous information. Research is currently being expanded to include carefully selected, uncluttered radar examples. The final goal of the investigation is the automatic identification of selected features from images acquired under a variety of conditions.

PREFACE

This report was prepared for the U.S. Army Engineer Topographic Laboratories, Fort Belvoir, Virginia 22060-5546, by Autometric, Incorporated, Falls Church, Virginia 22041.

The work was performed under the direction of the Contracting Officer's Technical Representative, Dr. Frederick W. Rhode, Team Leader, Center for Automated Image Analysis, and Mr. Lawrence A. Gambino, Director, Research Institute.

Colonel Alan L. Laubsher, CE, was Commander and Director, and Mr. Walter E. Boge was Technical Director of the Engineer Topographic Laboratories during the period of the investigation.

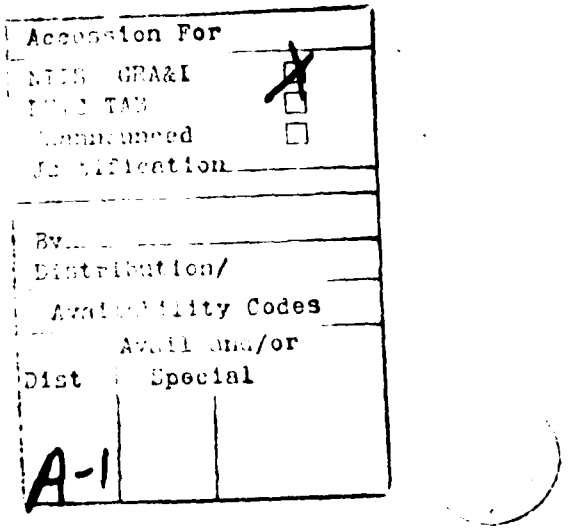


TABLE OF CONTENTS

Section No.	Title	Page No.
1.0	BACKGROUND AND INTRODUCTION	1
1.1	Previous Related Investigations	1
1.2	Current Investigation	6
2.0	OBJECTIVES	6
3.0	METHODOLOGY	7
3.1	Task 1 -- Validation of Descriptors and Descriptor Sets	
3.1.1	Subtask 1.1 -- Validation of Descriptors in Terms of Specificity and Reproducibility	7
3.1.2	Subtask 1.2 -- Validation of Descriptor Sets in Terms of Unique Feature Characterization	7
3.2	Task 2 -- Automated Extraction Using Computer Vision Techniques	12
3.2.1	Subtask 2.1 -- Generation and Organization of Digital, Black-and-White, Line-Drawn Images	12
3.2.2	Subtask 2.2 -- Adaptation and Development of Computer Vision Software that Detects and Identifies the 52 Descriptors	12
3.2.2.1	Raster Processing	12
3.2.2.2	Raster-to-Vector Conversion	14
3.2.2.3	Vector Processing	19
3.2.2.3.1	Consolidation of Feature Elements	19
3.2.2.3.2	Characterization of Primitives and Segments	21
3.2.2.4	Comparison of Derived Vector Data and Descriptor Sets	25
3.2.3	Subtask 2.3 -- Testing the Computer Vision Software	25
4.0	RESULTS	25
4.1	Raster Processing	27
4.2	Vector Processing	27
4.2.1	Point Features	27
4.2.2	Line Features	27
4.2.3	Area Features	28
4.3	Summary	28
REFERENCES		29

LIST OF FIGURES

Figure No.	Title	Page No.
1	Feature Descriptor Matrix for Roads	2
2	Boolean Expression and Number of Possible Descriptor Sets for the Feature "Roads"	4
3	Example of Rule-Based Classification: Roads	5
4	Four Illustrations of the Same Descriptor Set	8
5	Example of Feature Descriptor Matrix	9
6	Example Image Containing 16 Descriptor Sets from the Point, Line, and Area Categories	13
7	Set of Erosive Neighborhoods	15
8	A Group of Feature Pixels	16
9	Feature Pixels Identified as Belonging to Different Twigs	17
10	Classification of Feature Pixels Based on their Neighborhoods	18
11	Twig Showing "a", "b", "c", and "d" Feature Elements as Start, Merge, Split, and End Elements	20
12	Primitive Initially Containing Four Twigs	22
13	Three Point Segments	24
14	Flow Chart Used to Identify Point Descriptor Sets	26
A-1	Information Contained Within Header-Record of Segment File	A-10

LIST OF TABLES

Table No.	Title	Page No.
I	Requirement for Additional Descriptor Sets	11
A-I	Twig File Resulting from Raster-to-Vector Conversion	A-4
A-II	File Resulting from Grouping Twigs Into Primitives and the Removal of Split and Merge Elements	A-6
A-III	Primitive File After Topological Sorting is Complete	A-8
A-IV	Segment File Resulting from Analysis of Primitive File	A-9

APPENDIX A TWIG, PRIMITIVE, AND SEGMENT FILE FORMATS.....A-1
APPENDIX B SOFTWARE DESCRIPTION.....B-2

1.0 BACKGROUND AND INTRODUCTION

The research investigation described herein (Phase 3), identified and developed computer vision techniques that automatically recognize various graphic, line-drawn examples of feature descriptors that had been identified in preceding investigations as those that are used by image analysts for the interpretation of radar imagery. Fifty-two descriptors and 501 descriptor sets (feature identifiers) had been identified in the previous two phases of the project.

1.1 Previous Related Investigations

In these phases, descriptors were used to characterize features in image-space, leaving the identification to the image analyst. This top-down, goal-oriented approach to automatic feature extraction from radar imagery was inspired by an observation made by Rhode (1981) that states "The first step towards automated feature extraction (from images) is an in-depth analysis of the individual feature and the derivation of measurable, unique parameters that determine the feature unambiguously". This observation led to the Phase 1 and Phase 2 investigations by Pascucci and Huffman (1984) and Pascucci (1986), respectively.

In short, the objective of the Phase 1 and Phase 2 investigations was to develop the basis for an expert system that would consist of:

1) the compilation of feature descriptors based on the consensus of experienced image interpreters;

2) the development of sets of descriptors (feature identifiers) that relate to the detection, recognition, classification and identification of specific man-made and natural terrain features;

3) the establishment of a knowledge base containing the expert analysts' interpretational knowledge in the form of descriptors and condition/action rules for combining the descriptors into sets; and

4) an inference engine that applies if-then rules to the knowledge base so as to infer the identification of features from the descriptor sets.

SAR image examples of selected man-made features such as roads, railroads, bridges, and POL storage tanks were examined by experienced image interpreters in order to construct the knowledge base. The observed descriptors relating to a particular feature were presented in tabular format. Figure 1 is an example of the matrix that was developed for the feature "roads". While analyzing these matrices, it quickly became apparent that there might be more than one set of descriptors that characterizes a particular feature. In fact, in the example shown in Figure 1, the 12 roads that were examined exhibited four different descriptor sets.

ROADS

12	11	10	9	8	7	6	5	4	3	2	1	FEATURE NUMBER
4	3	3	2	2	2	1	1	1	1	1	1	SET NUMBER
												IMAGE REFERENCE NUMBER

aa	SINGLE POINT	NUMBER	POINT FEATURES	SPATIAL DESCRIPTORS
ab	GROUP OF POINTS			
ac	2 CLOSELY SPACED POINTS			
ad	IRREGULAR	PATTERN		
ae	LINEAR			
af	RECTANGULAR			
ag	RECTILINEAR	SHAPE (OF LINEAR PATTERN)		
ah	CURVILINEAR			
ai	COMPOUND			
aj	RECTILINEAR WITH ANGULAR BEND(S)			
ak	CIRCULAR/SEMICIRCULAR			
ba	SINGLE LINE	NUMBER	LINEAR FEATURES	
bb	GROUP OF LINES			
bc	2 PARALLEL LINES			
bd	3-4 PARALLEL LINES			
be	> 4 PARALLEL LINES			
bf	RECTILINEAR	SHAPE		
bg	CURVILINEAR			
bh	COMPOUND			
bi	RECTILINEAR WITH ANGULAR BEND(S)			
bj	CIRCULAR/SEMICIRCULAR			
bk	IRREGULAR	PATTERN		
bl	LINEAR			
bm	RECTANGULAR			
bn	PARALLEL TO DIRECTION OF RADAR	ORIENTATION		
bo	CONCAVE/CONVEX IN DIRECTION OF RADAR			
ca	SINGLE AREA	NUMBER	AREAL FEATURES	
cb	GROUP OF AREAS			
cc	RECTANGULAR	SHAPE		
cd	OVAL/CIRCULAR			
ce	POLYGONAL			
cf	IRREGULAR			
cg	LINEAR	PATTERN		
ch	RECTANGULAR			
ci	INTERLOCKING			
da	BRIGHT	BRIGHTNESS	INTENSITY DESCRIPTORS	
db	MEDIUM			
dc	DARK			
dd	NO-RETURN			
de	INTERMINGLED INTENSITIES	PATTERN		
ea	FINE	TEXTURAL DESCRIPTORS		
eb	MEDIUM			
ec	COARSE			
fa	LENGTH	DIMENSIONAL DESCRIPTORS		
fb	WIDTH			
fc	DIAMETER			
fd	SPACING			
ga	MEDIUM BRIGHTNESS	BACKGROUND DESCRIPTORS		
gb	LOW BRIGHTNESS			
gc	NO-RETURN, ONE SIDE			
gd	NO-RETURN, MORE THAN ONE SIDE			
ge	ON-LINE			

WRM108

Figure 1. Feature Descriptor Matrix for Roads

Another factor made apparent by the tabulation of the descriptor sets is that all of the observed descriptor sets can be expressed in Boolean notation as a single formulation, as shown in Figure 2. Further, the solution of the Boolean expression tells us that, given these individual descriptors and no others, there can be no more than six possible descriptor sets for roads (two "numbers", multiplied by three "shapes", multiplied by one "brightness"). This formulation, either in English language or in Boolean, leads to the rule-based classification scheme shown in Figure 3.

The work continued into Phase 2 where the objectives were:

1) to validate the descriptor sets that were developed in Phase 1;
2) to develop and validate additional descriptor sets; and
3) to integrate the validated Phase 1 and Phase 2 descriptor sets into a computer-assisted expert system for classifying features on SAR imagery (Pascucci, 1986).

A statistical analysis was conducted to determine what constitutes an adequate sample size of imagery. Initially, it was determined that 12 images are required for an adequate sample size for any particular feature. Features for which fewer than 12 image examples could be found were not used. However, as the work progressed, it was determined that, as the sample size increases above 12, additional descriptor sets may be found. Thus, the adequate sample size must be regarded as open-ended, and it is therefore necessary to continuously update the knowledge base of the system.

During Phase 2, descriptor sets were developed for 15 additional man-made and natural features, bringing to 29 the total number of features that could be identified using descriptors sets. Also, the total number of descriptors was increased from 39 (found in Phase 1) to 52.

A computer program was also written that decomposes complex Boolean expressions and automates the classification of features from any of the input descriptor sets. Each complex Boolean expression comprises all of the descriptor sets that characterize a single feature, and when these expressions are decomposed, each descriptor set is explicitly listed, making automatic identification possible and verifying the uniqueness of the descriptor set.

Validity of the descriptors and descriptor sets was tested using untrained image anlysts. The reason for using untrained analysts was to ensure that identification of features would be made only on the basis of the descriptor sets with which they were provided, without recourse to additional knowledge that the experts may use without realizing it. The analysts were given instruction in the recognition of the descriptors and in the condition/action rules that govern the combination of individual descriptors into sets. These analysts were then tested on 138 SAR image examples of the 52 descriptors. Because only two analysts were available for this training, and because their test scores of 93% and 78% were so disparate, the verification was inconclusive. However, since most of the errors made by the analysts were due to fatigue, carelessness and forgetfulness -- sources of error that are not experienced with computer vision systems -- eventual validation appeared promising, and the work continued into Phase 3, which is documented in this report.

ENGLISH LANGUAGE EXPRESSION

SINGLE LINE OR TWO PARALLEL LINES, AND RECTILINEAR SHAPE OR CURVILINEAR SHAPE OR COMPOUND SHAPE, AND NO-RETURN BRIGHTNESS.

BOOLEAN EXPRESSION

$(ba \cup bc) \cap (bf \cup bg \cup bh) \cap dd$

NUMBER OF POSSIBLE DESCRIPTOR SETS

$(1+1) \times (1+1+1) \times 1 = 6$

WRM134

Figure 2. Boolean Expression and Number of Possible Descriptor Sets for the Feature "Roads".

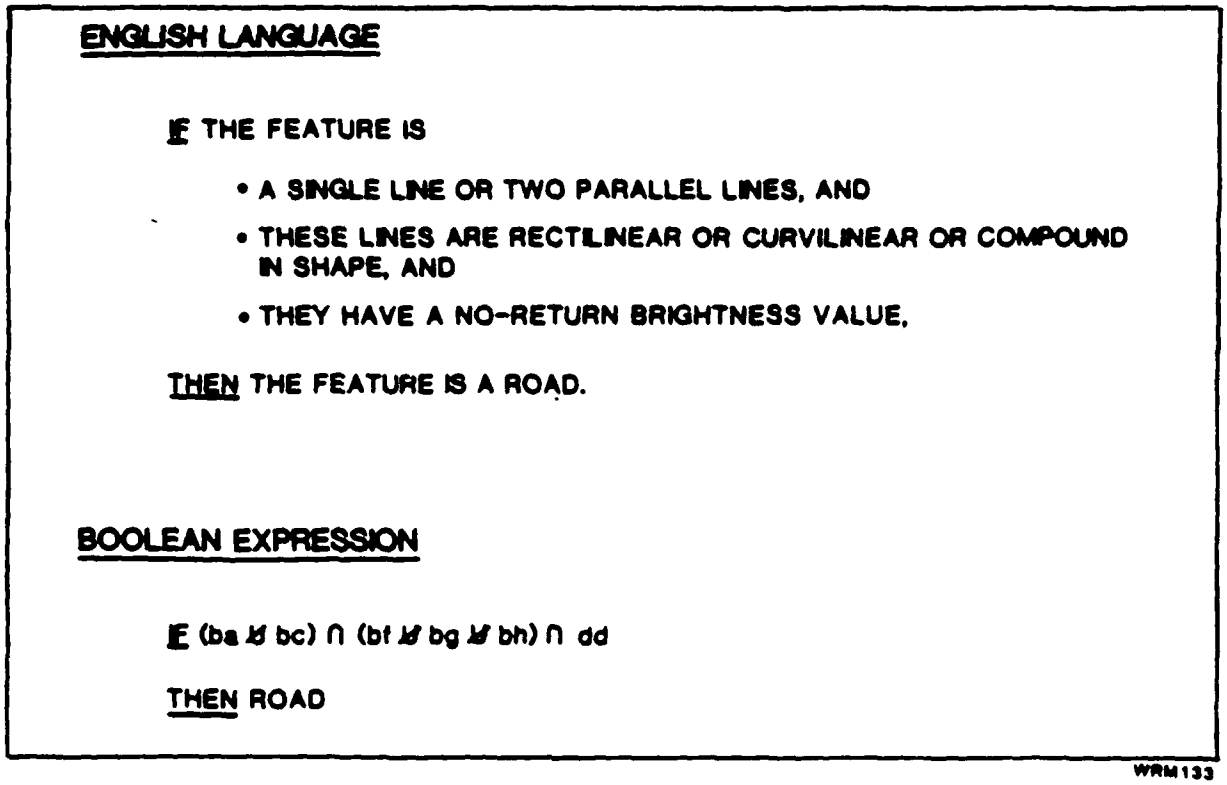


Figure 3. Example of Rule-Based Classification: Roads

1.2 Current Investigation

Phase 3 was designed to carry the automated process of feature identification a step further by developing software that could replace the image analyst in his remaining function of descriptor identification.

The automated descriptor identification procedure includes four steps: 1) raster processing (e.g., edge enhancement, density slicing, neighborhood encoding and thinning); 2) raster-to-vector conversion; 3) processing of the resultant vector data (e.g., identification of points, lines and areas, referred to as primitives, including a description of primitive size, shape, position and orientation); and 4) the examination of the relative positions of primitives including the determination of the similarity between groups of primitives and descriptor sets.

The work was conducted at the ETL Laboratory for Automated Radar Image Analysis on a VAX 11/780 computer using the ULTRIX-32 operating system. Software was developed in C, and a Grinnell image processing system was used for image display.

2.0 OBJECTIVES

The overall objectives of this phase of the investigation are:

- Objective A - Update and validate descriptor sets using test results from USAETL and the U.S. Army Intelligence Center and School (USAICS); and
- Objective B - Adapt, develop and apply computer vision techniques to the verified descriptors as a further step toward a completely automated expert system for SAR image feature recognition.

These objectives were achieved by carrying out the following tasks and subtasks:

- Task 1 — validate and update existing descriptors, feature identifiers and rules;
- Task 2 — extract features automatically using computer-vision techniques;
- Subtask 2.1 -- generate and organize digital graphic, line-drawn images;
- Subtask 2.2 -- adapt and develop computer-vision software that detects, recognizes and identifies descriptors of features on digitized graphic examples of the 29 features developed in Phase 1. The software shall be developed in the C Language and will be capable of running on the VAX, 11/780 computer using the ULTRIX-32 operating system;
- Subtask 2.3 -- test and demonstrate the computer-vision software at ETL on a VAX 11/780 computer and the Grinnell image processor using images approved by USAETL.

3.0 METHODOLOGY

The following sections describe the purpose and method of execution of the above tasks.

3.1 Task 1 — Validation of Descriptors,and Descriptor Sets

3.1.1 Subtask 1.1 — Validation of Descriptors in Terms of Specificity and Reproducibility

Black and white, hand-drawn graphic representations of 52 sets of descriptors were prepared for the purpose of testing their validity in terms of the specificity and reproducibility of their denotation by untrained image analysts or by a computer vision system. For example, all four of the illustrations in Figure 4 are described by expert radar analysts as follows:

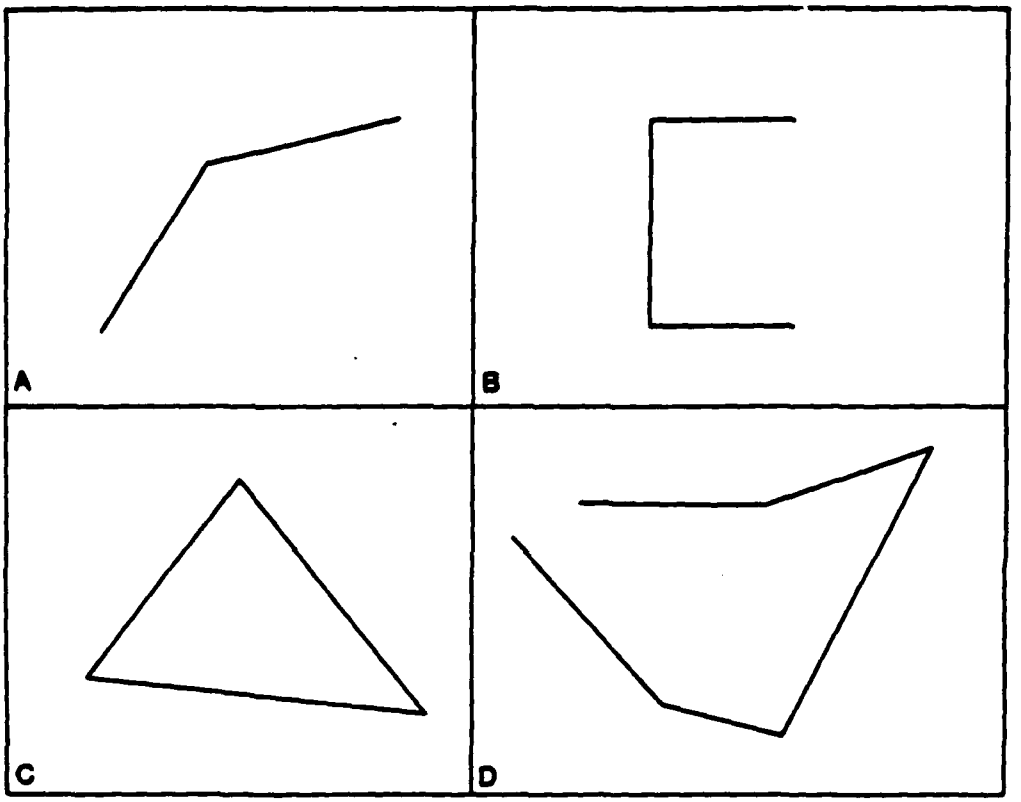
Type of Descriptor	-- linear feature
Number	-- single line
Shape	-- rectilinear with angular bend(s)
Brightness	-- no return (black)

If the descriptors that make up these four illustrations have been defined with sufficient rigor and clarity, and if these descriptions have been thoroughly "learned" by untrained analysts and by a computer-vision system, then humans and machines alike should produce the same description as that produced by the expert analysts. To test this, the 52 sets of descriptors were bound into test books along with a list of the definitions of the descriptors and with at least one graphic example of the appearance of each descriptor. The test books were delivered to the U.S. Army Intelligence Center and School at Fort Huachuca, Arizona, which distributed them to 55 students on the first day of the radar portion of the school's Image Analysis course. The students were given four hours to familiarize themselves with the definitions and examples of the descriptors and then to describe each of the 52 descriptor sets in the test book by filling out a Feature Descriptor Matrix (see Figure 5).

The test results, while favorable, were inconclusive, the principal reason being that the test administrators were unable to determine whether an error had been caused by an ambiguous definition of a descriptor or by incomplete understanding of the definition by the student. It appears that this lack of conclusiveness will persist until definitions are incorporated into the software of the completed computer vision system.

3.1.2 Subtask 1.2 -- Validation of Descriptor Sets in Terms of Unique Feature Characterization

Because the 29 simple features with which this investigation has thus far dealt can be identified unambiguously and without error by expert image analysts, it follows that each of the 501 descriptor sets that have been found to characterize these features is unique and denotes one feature only. For example, if two or more of the 29 features were to have a descriptor set in common, the expert analysts would occasionally misidentify a feature or be unsure of its identity.



A, B, C, & D ARE ALL EXAMPLES OF A LINEAR FEATURE, CONSISTING OF A SINGLE LINE, HAVING A RECTILINEAR SHAPE WITH ANGULAR BEND(S), AND A NO-RETURN (BLACK) BRIGHTNESS.

WRM124

Figure 4. Four Illustration of the Same Descriptor Set

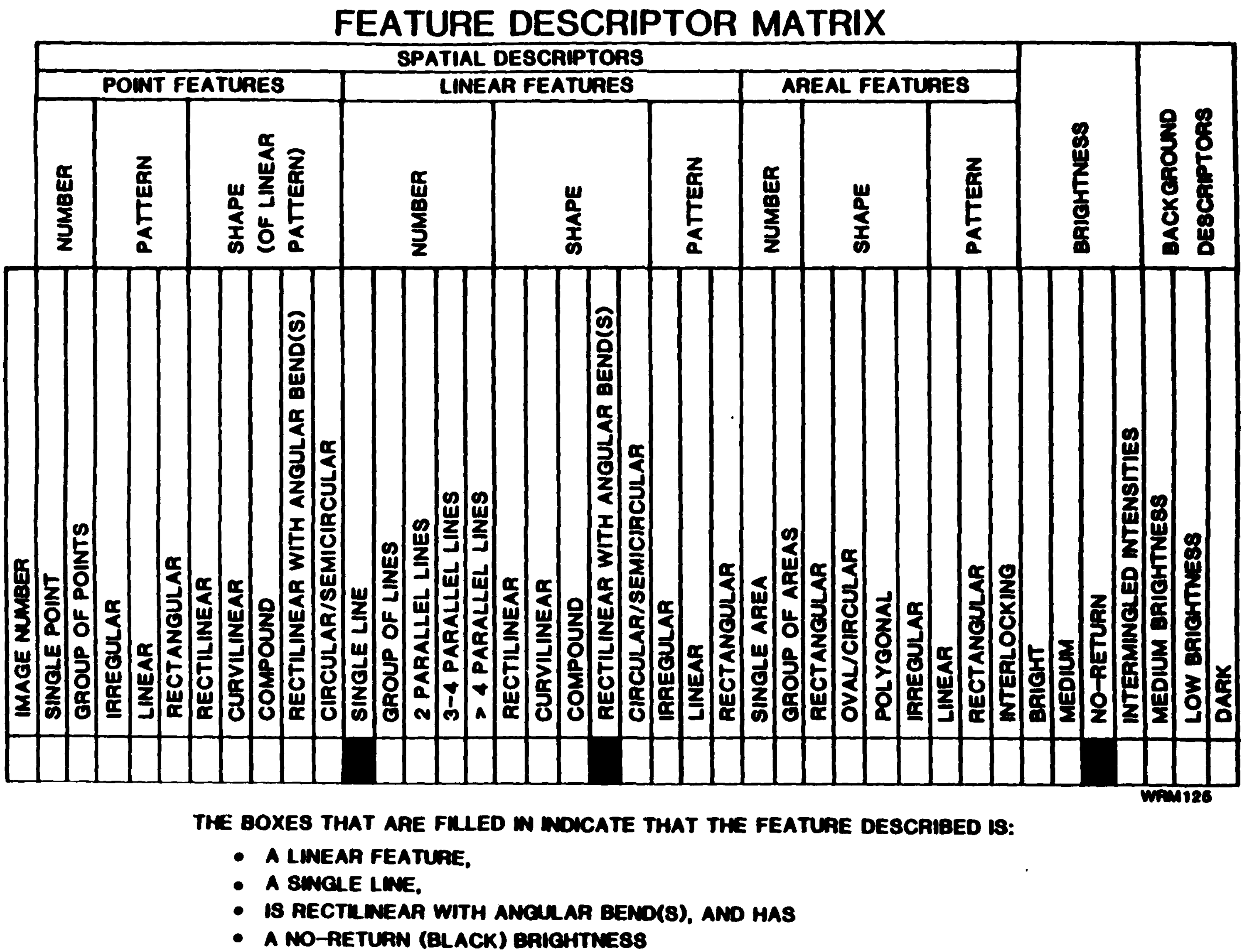


Figure 5. Example of Feature Descriptor Matrix

To test whether or not each descriptor set uniquely identified a single feature, the expert system was instructed to produce a printout, in alphabetical order, of each of the 501 descriptor sets, along with the feature that it characterizes. Looking at the printout, the investigators found nine sets of features that share a common descriptor set (i.e., sets of features that can be mistaken for each other under certain circumstances):

- dams and piers
- dams and breakwaters
- breakwaters and causeways
- canals and runways
- roads and runways
- breakwaters and piers
- canals and roads
- canals and rivers
- rivers and roads

Since expert analysts never mistake dams for piers and are never unsure whether the feature at which they are looking is a breakwater or a causeway, it follows that the 52 descriptors that are being used in the knowledge base of the expert system may be necessary, but are not sufficient, for unambiguous identification. Therefore, additional descriptors have to be identified. It is clear that the missing descriptors are used by the expert analysts in identifying features, otherwise they would make mistakes. It is equally clear that the experts were not aware that they were using these descriptors, otherwise they would have identified them and added them to the knowledge base.

After it had been pointed out that additional descriptors would be necessary for the unambiguous identification of the nine sets of features that have descriptor sets in common, the expert analysts examined the sets and determined the required descriptors, which are shown in Table I. As shown in the table, the ambiguity of five of the sets can be resolved by introducing two new descriptors: no return at one end; and low return at both ends. However, for the remaining four sets of features, the ambiguity can be resolved only by looking at associated features. This requirement transforms the five features in the remaining four sets (breakwaters, piers, canals, roads, and rivers) from "simple" to "complex" features. Throughout the foregoing phases of the investigation, the working definition of a "simple" feature has been "a single feature that can be fully characterized by its own sets of descriptors". "Complex" features, on the other hand, are "features that are composed of more than one simple feature and that can be fully characterized only by the descriptor sets of all of its component simple features". An example of a complex feature would be an oil refinery, which might consist of docks, POL tanks, rail tracks, and service roads. Thus far, the process of grouping related simple features into complex features has not been undertaken, and it is the opinion of the investigators that it probably should not be undertaken until the computer vision techniques have been further developed and implemented. At that time, when we have a better knowledge of the strengths and limitations of the techniques and are more familiar with their applications, the problems associated with the grouping of related features can be analyzed.

TABLE I.

REQUIREMENT FOR ADDITIONAL DESCRIPTOR SETS

AMBIGUOUSLY DEFINED FEATURE SETS	DESCRIPTORS NEEDED TO RESOLVE AMBIGUITY
DAMS & PIERS	NO RETURN, ONE END
DAMS & BREAKWATERS	NO RETURN, ONE END
BREAKWATERS & CAUSEWAYS	NO RETURN, ONE END
CANALS & RUNWAYS	LOW RETURN, BOTH ENDS
ROADS & RUNWAYS	LOW RETURN, BOTH ENDS
BREAKWATERS & PIERS	ASSOCIATED FEATURES (TRACKS, WAREHOUSES, ETC.)
CANALS & ROADS	ASSOCIATED FEATURES (INTERCHANGES, LOCKS, ETC.)
CANALS & RIVERS	ASSOCIATED FEATURES (TRIBUTARIES, LOCKS, ETC.)
RIVERS & ROADS	ASSOCIATED FEATURES (BRIDGES, INTERCHANGES, ETC.)

WRM131

3.2 Task 2 — Automated Feature Extraction Using Computer Vision Techniques

The feature extraction procedure includes raster processing, raster-to-vector conversion, and processing of the resultant vector data. The raster processing stage consists of an enhancement of local spatial variation to identify edges and texture. In addition, a thresholding and thinning operation was performed to eliminate non-essential feature pixels. After essential feature pixels had been identified in the raster image, they were converted to vector format. This was done so that individual groups of feature pixels could be analyzed separately. Finally, after feature pixels had been restored in vector format, they were first consolidated, again to remove non-essential information, and then characterized with respect to their position, shape, orientation, etc. in order to more easily compare them with descriptor sets that were identified in the early phases of the study.

3.2.1 Subtask 2.1 — Generation and Organization of Digital, Black-and-White, Line-Drawn Images

Eight-bit digital images of 42 descriptor sets were created using a Hamamatsu video digitizing system attached to an HP-1000 computer and a tape drive. The scale of the line-drawn images was selected so that a single descriptor could be displayed in a 128 x 128 pixel subsection of a 512 x 512 pixel color monitor. This allowed simultaneous analysis of 16 different descriptor sets. Descriptor sets were grouped into three major categories: points, lines and areas. Three images were then created, each containing 16 examples from a particular category, along with a fourth image in which all three categories of descriptor sets were combined (Figure 6).

3.2.2 Subtask 2.2 — Adaptation and Development of Computer-Vision Software that Detects and Identifies the 52 Descriptors

3.2.2.1 Raster Processing

The first operation performed on the raster images was an enhanced local spatial variation using a 3 x 3 filter. The center pixel in the output image was assigned the sum of the absolute differences between the center pixel and each of its surrounding neighbors in the original image as shown in the equation.

$$'P_{I,J} = \sum_{m=1,\,n=1}^{3}\sum^{3} \left| P_{I,J} - P_{I-2+m,\,J-2+n} \right| \qquad (1)$$

This produced an image in which pixel brightness was directly proportional to local spatial variation. A histogram was then generated of the enhanced image, and a threshold was chosen that separated the brightest edges from the rest of the scene. The selection of the threshold was based on the number of brightness categories that were used by the analyst and on the minimum brightness of an edge occurring between two adjoining brightness categories.

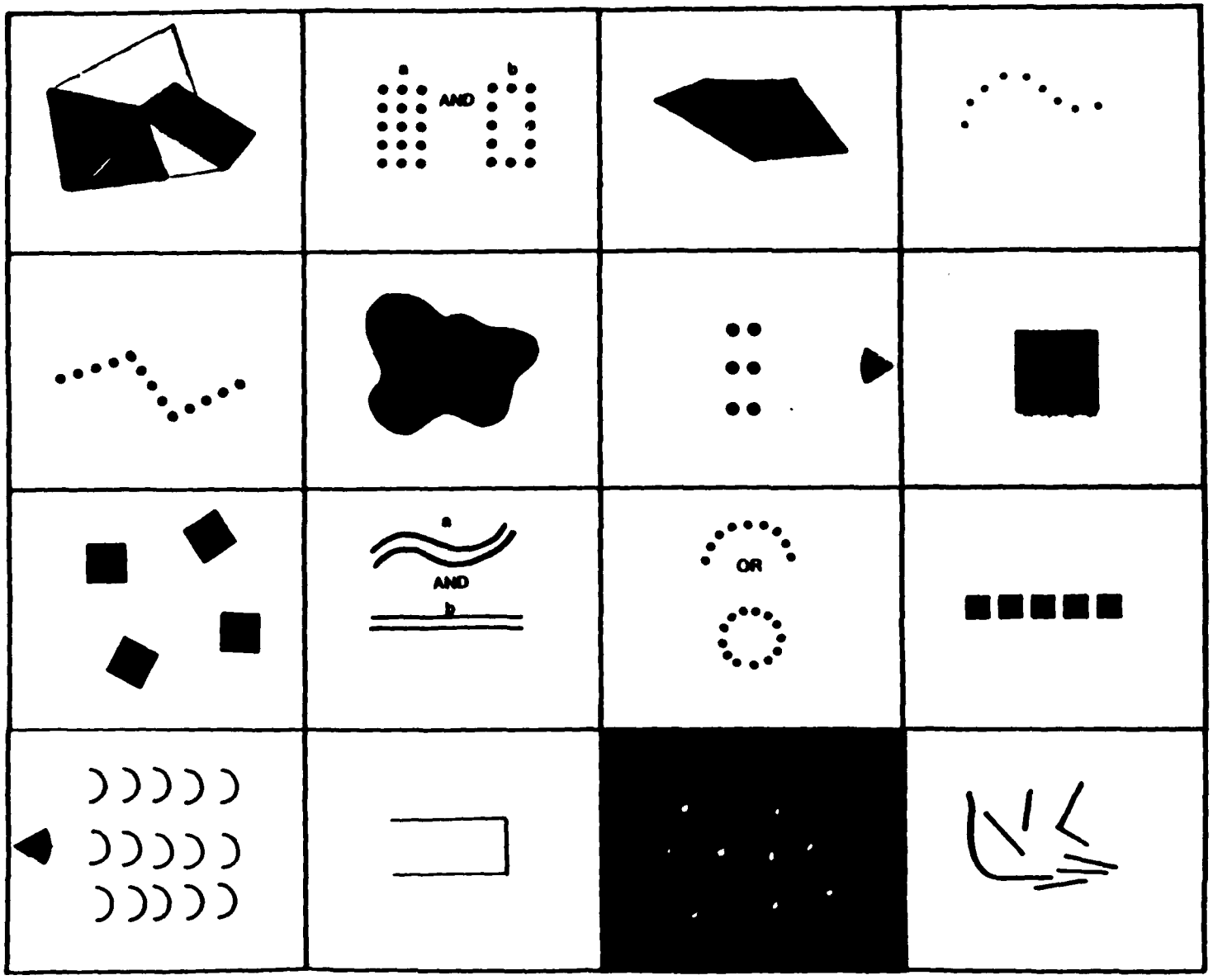


Figure 6. Example Image Containing 16 Descriptor Sets from the Point, Line, and Area Categories

The selection of the threshold was less critical than it would be under normal circumstances because the topological sorting that is performed later in the vector processing phase reduces much of the extraneous information, thus decreasing the sensitivity of the procedure to the selection of an artificially low threshold (Speck, 1980).

As a final step, the new binary image was processed following a procedure outlined by Woetzel (1978) in which the image was neighborhood encoded and then thinned using a connectivity-preserving algorithm. The resultant binary image contains background pixels equal to zero and feature pixels equal to one.

A filter, shown in the equation

$$A = \begin{bmatrix} 8 & 4 & 2 \\ 16 & 0 & 1 \\ 32 & 64 & 128 \end{bmatrix}$$

$$\text{NEIGHBORHOOD CODE FOR } P_{I,J} = \sum_{m=1,\, n=1}^{3} \sum^{3} a_{m,n} \cdot P_{I-2+m,\, J-2+n} \qquad (2)$$

was passed over the binary image, and a new neighborhood-encoded image was created. The filtering operation can be performed at video rates using a standard array or feedback processor available on many image processing systems. Pixel values in the neighborhood-encoded image range from 0-255, each specifying a different neighborhood with all possible neighborhoods defined. The image was then searched and feature pixels with particular neighborhoods eliminated. Woetzel (1980) presented the set of erosive neighborhoods shown in Figure 7. Also, it was necessary to modify triangular junction points in order to remove unnecessary information. For example, in Figure 8, feature pixels "a" and "b" have neighborhoods with brightness values of 145 and 82, respectively. The essential topological information is not lost if the link between feature pixel "a" and its lower right-hand neighbor is removed. This gives "a" a new neighborhood code of 17, which is a valid line element. Without removal of the non-essential information, identification of "c" as a member of only twig number three is not clear (Figure 9).

Raster processing of radar imagery (as opposed to the processing of black-and-white graphics) will probably follow a procedure similar to the one outlined above. A possible addition may be the insertion of an edge-preserving, smoothing operation at the beginning of the raster-processing operation. This would have the effect of removing unwanted high-spatial-frequency noise (i.e., radar speckle).

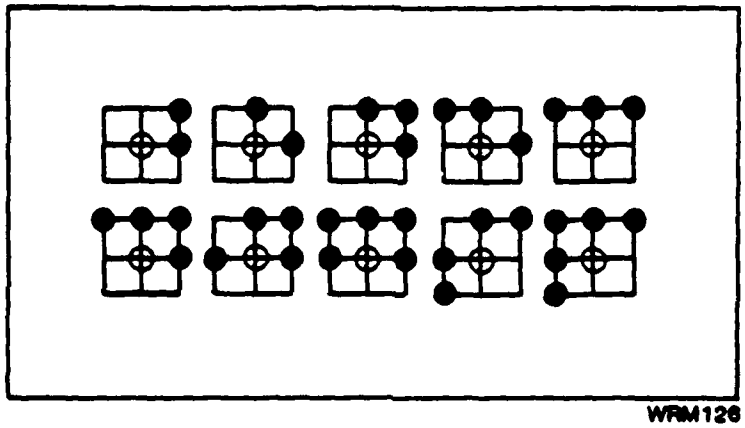


Figure 7. Set of Erosive Neighborhoods -- symmetric neighborhoods are not shown (Woetzel, 1978).

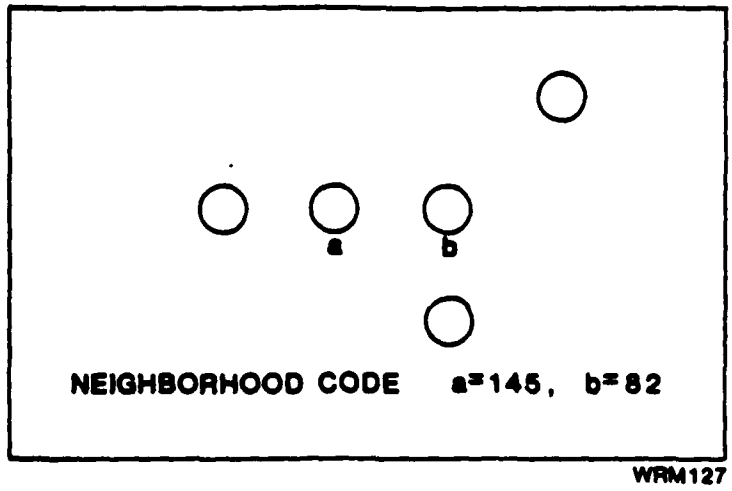


Figure 8. A Group of Feature Pixels, where "a" is a line element and "b" is a knot element

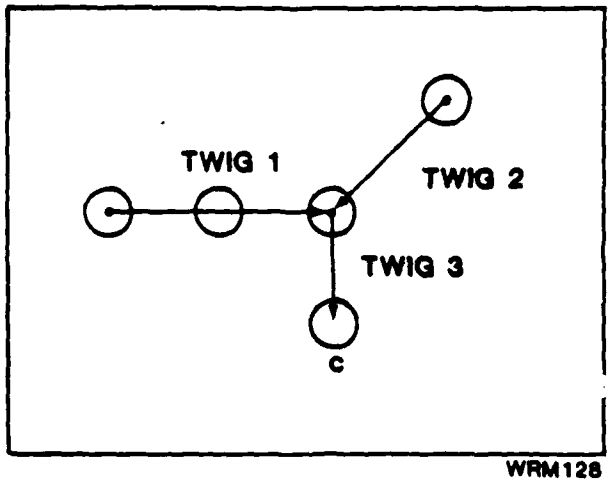


Figure 9. Feature Pixels Identified as Belonging to Different Twigs

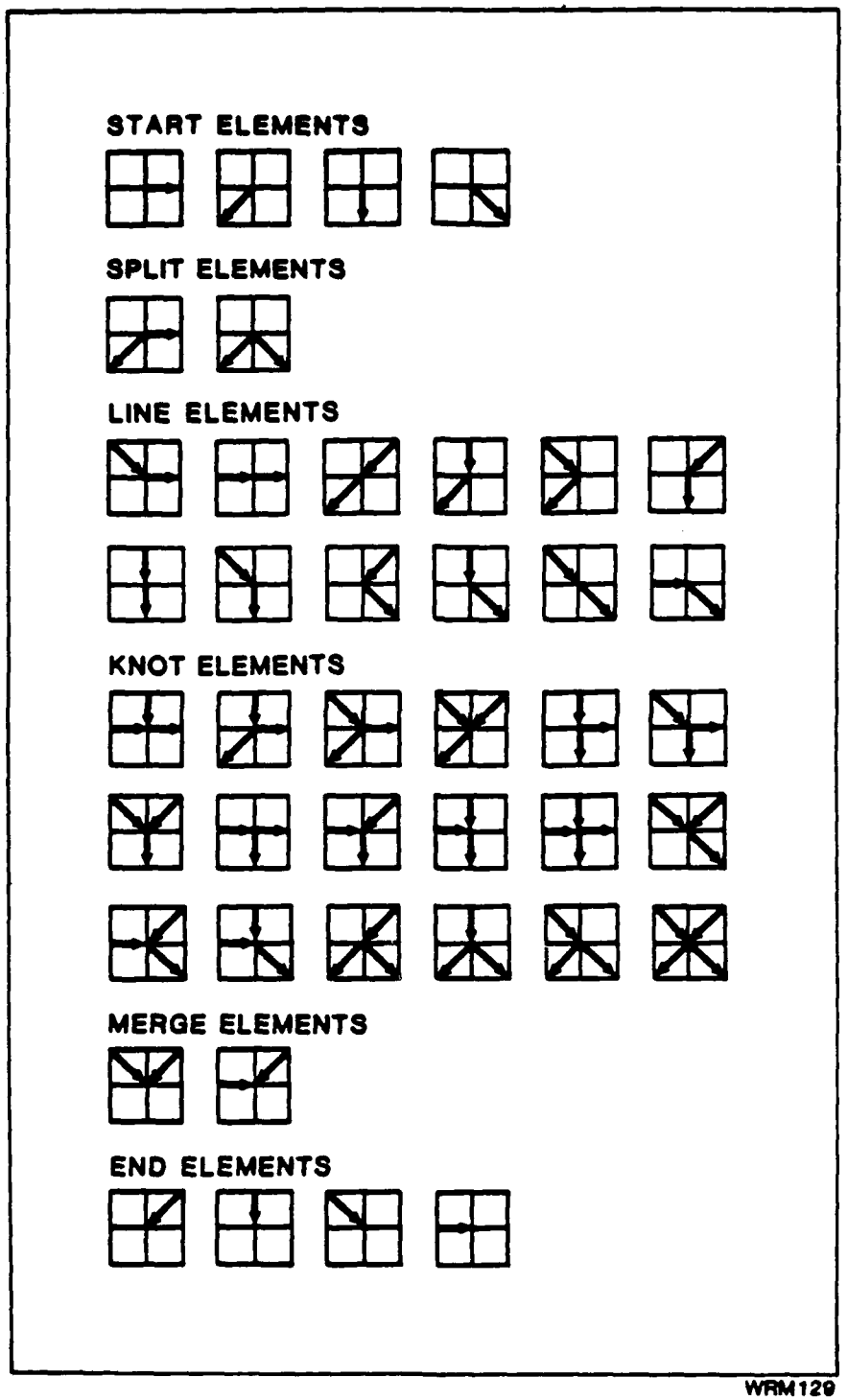


Figure 10. Classification of Feature Pixels Based on their Neighborhoods (Speck, 1980)

3.2.2.2 Raster-to-Vector Conversion

A raster-to-vector conversion algorithm was applied to the thinned image in a method similar to that discussed by Speck (1980). A rake, the width of the image, was passed over the image from top to bottom with all feature elements being identified as start, split, line, knot, merge or end elements, depending on their neighborhood (Figure 10). When a start or split element was encountered, a new feature element series was started. These groups of feature elements are referred to as twigs. Knowing the neighborhood of a feature element in a particular twig, the investigator can easily determine the next position in the rake in which the twig appears, and a pointer can be stored there identifying the twig to which that feature pixel belongs. When a knot was encountered, the appropriate twigs were terminated or begun.

When the raster-to-vector conversion is complete, a twig file exists that contains: the number of twigs in the file, an identifying twig number, the number of elements in each twig, the x,y coordinates for each element in each twig, and the type of the first and last element in the twig (see Appendix A).

3.2.2.3 Vector Processing

3.2.2.3.1 Consolidation of Feature Elements

Once the raster image had been reduced to vector format, many algorithms were developed to sort, remove and group feature pixels into objects that could be matched with the descriptors defined earlier in the study. Feature pixels were grouped together in three stages as twigs, primitives and segments.

In the first step, feature pixels were grouped into twigs, as discussed in the raster-to-vector conversion section. The first element of a twig can be a start element, a split element or a knot element, and the last element of a twig can be an end element, a merge element or a knot element. Initially, the topographical sorting changed the merge and split element to line elements by combining the two series of elements that started or ended at the split or merge element, respectively. These elements are simply line elements but were not originally classified as such owing to their position in the original image as shown in Figure 11. Feature element "b" has a neighborhood code of 18, which identifies it as a merge element, and "c" has a neighborhood code of 33, which identifies it as a split element. Again, classification of feature elements as start, split, line, knot, merge and end elements is outlined in Figure 10. After removal of split and merge elements, twigs were differentiated on the basis of connectedness as outlined by Speck (1980):

strips - no connection at either end
hairs - one end connected to another twig
joints - both ends connected to other twigs
curls - twig closing on itself (i.e., a point or an area).

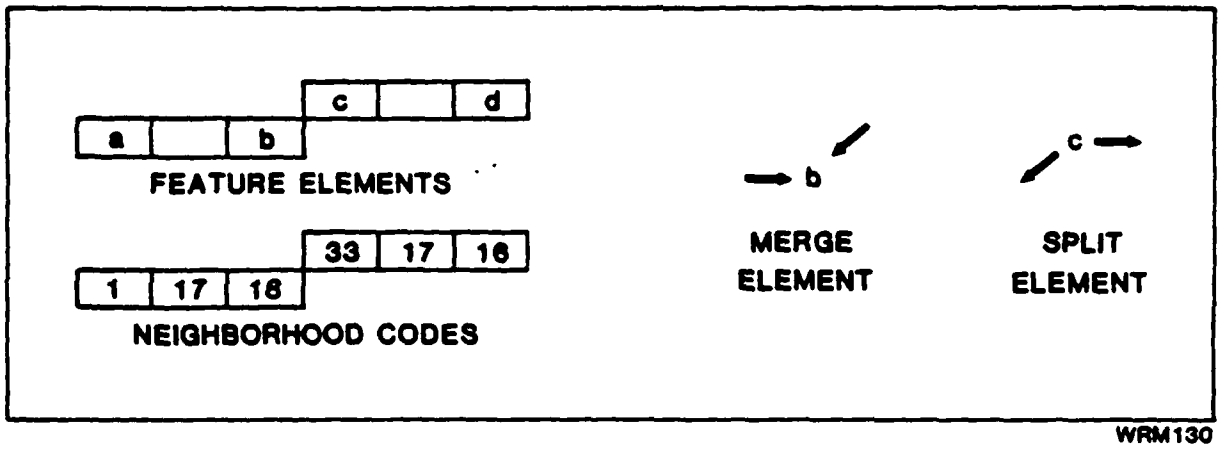


Figure 11. Twig Showing 'a', 'b', 'c', and 'd' Feature Elements as Start, Merge, Split, and End Elements, Respectively. 'c' is the First element encountered during the top-down search in the raster-to-vector conversion. It is initially identified as a split element and must be reclassified as line element.

At this stage, two levels of consolidation were added to Speck's approach. First, twigs were grouped into primitives, which were defined as connected twigs. For example, the primitive in Figure 12a is made up of four twigs; two joints and two hairs. Any point in the primitive can be reached from any other point without "stepping off" a feature pixel.

Once twigs were grouped into primitives, additional sorting was required to further consolidate twigs within a primitive. After additional sorting, the primitive in Figure 12b can be viewed as being made up of a curl and two hairs. This consolidation not only became important when comparing groups of twigs to descriptor sets, but also simplified the polygon approximation procedure that is required for differentiating various line types. An example of a primitive file before and after sorting is shown in Appendix A.

During the third consolidation phase, primitives were grouped into segments based on their proximity to each other. Currently, in order to simplify the operation, primitives in the same 128 x 128 pixel subsection are considered to be in the same segment. When analysis of radar imagery begins, and image scale is provided, areas surrounding primitives will be searched for other primitives close enough to be considered in the same segment. The maximum allowable distance between primitives for them to be included in the same segment will be determined empirically for various descriptors and used during this final consolidation phase. Segments are at the same level of consolidation as descriptors and are therefore directly comparable. A segment file is shown in Appendix A.

3.2.2.3.2 Characterization of Primitives and Segments

The first step in identifying the characteristics of groups of feature pixels was the generation of statistics for individual primitives. Coordinate means and variances, slope, intercept, eigenvalues and eigenvectors (from coordinates of feature pixels making up primitives) were then used to describe primitive size, shape, position and orientation in image space.

These statistics were first used to identify simple shapes, such as points and straight lines, early in the topological sorting process. This reduced the number of primitives that the investigator would otherwise have had to run through the full analysis procedure. After initial analyses and topological sorting were complete, primitive files containing coordinates of feature elements making up the primitive were passed to a polygon approximation routine. The polygon approximation routine follows the procedure outlined by Pavlidis (1982). The method, however, has one shortcoming in that it forces the approximation through the curve endpoints and consequently does not necessarily produce either maximum length vectors or optimally smoothed results. The algorithm was modified to reduce the effects of this constraint by checking that the length of the first and last line segments are above a predefined minimum. If one of the end line segments is short and was created just so the approximation would pass through an outlying endpoint, then it is removed. If this polygon approximation routine is recognized as being a limiting factor as the work progresses, other more sophisticated algorithms (Williams, 1978; 1981), will be investigated. The resultant polygon data were then used to distinguish between lines with angular bends, curvilinear lines, and compound lines and to determine the shape of areas.

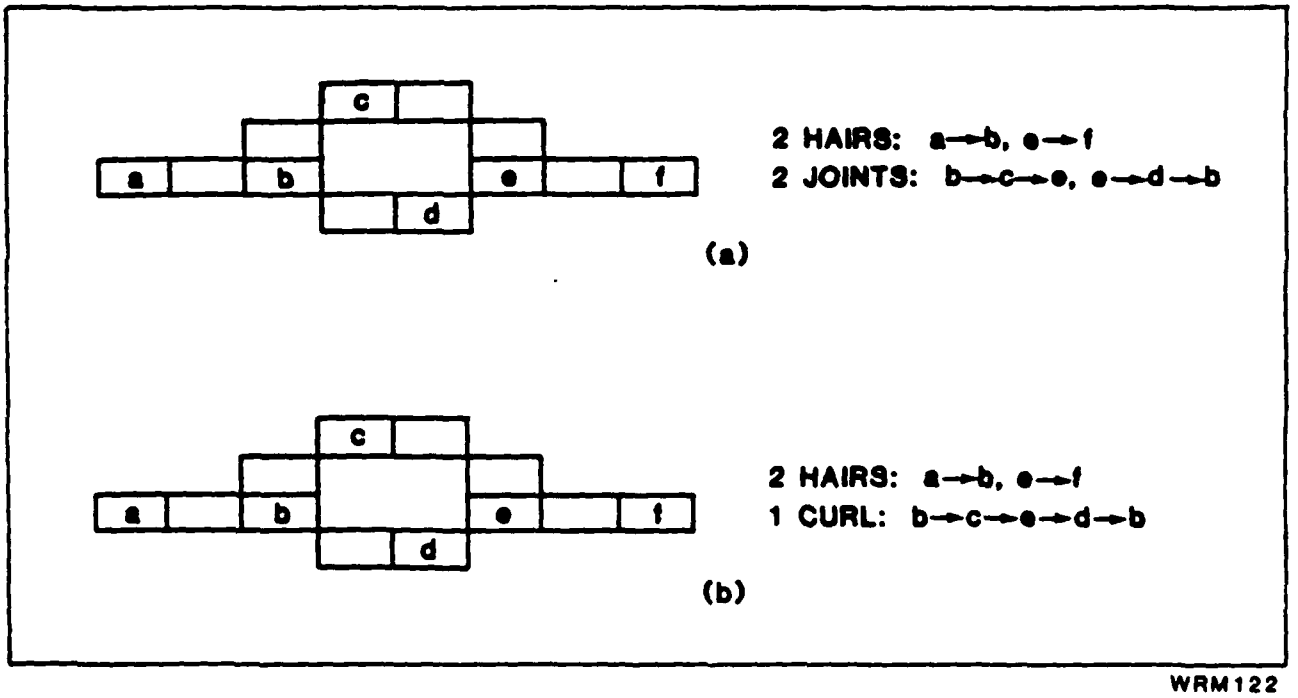


Figure 12. (a) Primitive Initially Containing Four Twigs: Two Hairs and Two Joints.

(b) Primitive After Twig Consolidation Containing Three Twigs: Two Hairs and One Curl.

After all primitives had been identified, the primitive statistics that had been generated earlier were used to group them into segments and to identify segment patterns. As stated earlier, in order to simplify the operation during Phase 3, primitives in the same 128 x 128 pixel subsection were considered to be in the same segment.

When primitives were grouped into segments, a segment file was created that contains the number of primitives in the segment, the number and types of twigs contained in each primitive, the number of feature pixels per twig, and the coordinates of the feature pixels. A header record was also created in which all the information required to make comparisons between segments and descriptor sets was stored. The header record contains information described in Figure 12.

After the primitives had been grouped into segments, primitive positions, shapes and orientations were used to identify patterns within the segments. For example, Figure 13 contains three segments that have been identified on the basis of the angles that are formed by any three adjacent primitives and the statistics that describe this group of angles. This was accomplished by sorting primitive means so that the algorithm can be moved from one end of the line to the other. The angle formed by primitive 1, 2, 3 was calculated, then 2, 3, 4 and 3, 4, 5, etc. The average and variance of the angles that are not within a specified range of 180 degrees were calculated. It was not necessary to consider straight segments in order to determine if a pattern was curvilinear or angular. In the case of segment 1, no non-180-degree angles exist, and the pattern was classified as a straight linear pattern. For segment 2, the angles not falling into the 180-degree category were found to be more acute than those measured in segment 3. Because the number of primitives in a segment is relatively small when compared to the number of feature elements in a primitive, it was possible to employ this somewhat inefficient method of pattern determination.

In the analysis of line patterns, orientation was a significant characteristic and was used to identify parallel lines and line orientation with respect to the radar. In this phase of the study, the radar beam was arbitrarily chosen to be coming from the top of the display screen, which was also designated as north. Objects were considered to be parallel to the beam of the radar if the major axis was within a range specified earlier by the image analyst.

Areas and area patterns were treated in a manner that was similar to the treatment of line and line patterns. The major differences were that areas have a texture descriptor, and areas of different texture or brightness could be contiguous, forming an interlocking pattern that complicated the topological sorting scheme.

Finally, when primitives were identified and patterns determined, the original image file was reopened and accessed along with the texture-enhanced image in order to determine primitive brightness, texture and the brightness of the background. Brightness and texture were easily determined for points and areas by locating interior pixels using primitive statistics and edge information from the thinned image. In the case of lines, movement of the edges during top-down, left-to-right thinning had to be considered when determining location of the line in order to measure line and background

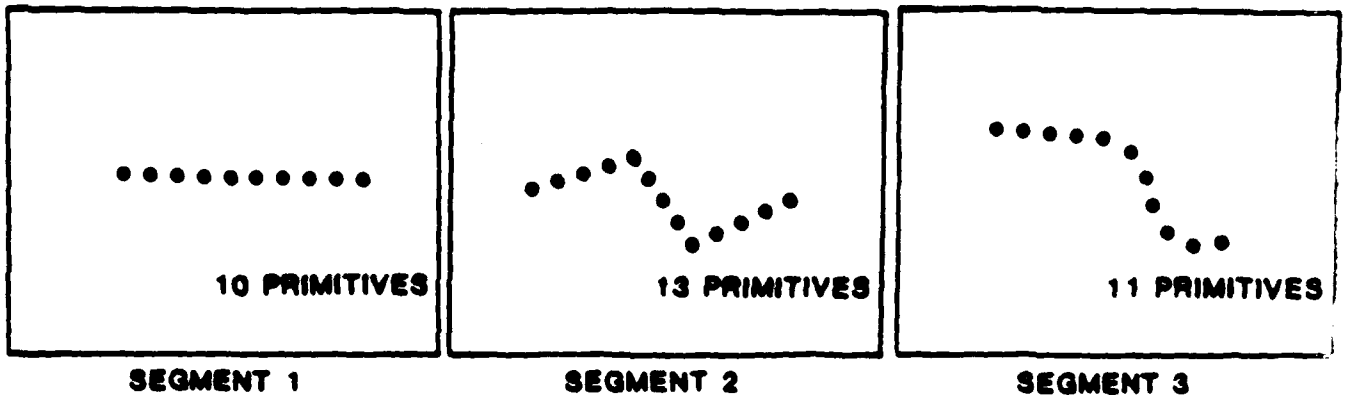


Figure 13. Three Point Segments Identified as a Straight Linear Pattern, a Linear Pattern with Angular Bends, and a Curvilinear Pattern, Respectively.

brightness. Five brightness levels were chosen: dark, medium-dark, medium, medium-bright, and bright. These categories were equally spaced from 0-255. When analysis of radar imagery begins, a method of radiometric normalization will have to be considered to allow for completely automatic feature extraction. Texture was the most difficult characteristic to simulate, with a great deal of the detail lost during digitization. Texture categories were left at fine, medium and course, but these descriptors will be quantified when the analysis of actual radar imagery begins.

Overall, data were generated that allow traversal through a flow chart similar to the one shown for points in Figure 14.

3.2.2.4 Comparison of Derived Vector Data and Descriptor Sets

Currently, a temporary routine is used that functions similarly to the program that was developed in Phase 2, which decomposes complex Boolean expressions and automatically classifies features from any input descriptor set. This program is simply a series of nested if-then-else statements that reads header information from the segment file and displays descriptor information in English for use during debugging. In the future, this routine will be replaced by another program that will act as an interface between the computer-vision software developed during this phase and the expert system developed in the previous phase. This interfacing program will read information from the header record in the segment file and produce the information required by the expert system.

3.2.3 Subtask 2.3 -- Testing the Computer Vision Software

Software was developed in a modular format in order to allow testing of different functions at different stages. Each module output two disk files, an ascii-format file that contains diagnostic output, and a binary file in a fixed format, which contains information required by the next module. This allows the programmer access to detailed diagnostic information at every stage of execution.

4.0 RESULTS

The results of Subtask 1.1 -- Validation of Descriptors in Terms of Specificity and Reproducibility -- while favorable, were inconclusive. This was because of the difficulty in determining whether the errors in assigning descriptors to an imaged descriptor set were caused by shortcomings in the definitions of the descriptors or by an incomplete understanding of the descriptor definition by the untrained analysts. This lack of conclusiveness will persist until the descriptor definitions are incorporated into the computer vision software and tested.

The results of Subtask 1.2 -- Validation of Descriptor Sets in Terms of Unique Feature Characterization -- showed that, in order to resolve ambiguous identification, two additional descriptors must be added to the 52 that are currently in the knowledge base of the expert system. A more important result was that five features - breakwaters, piers, canals, roads, and rivers - cannot be characterized unambiguously on the basis of their own descriptor sets but require the descriptor sets of associated features for unambiguous identification. This problem will be addressed in the next phase of the investigation.

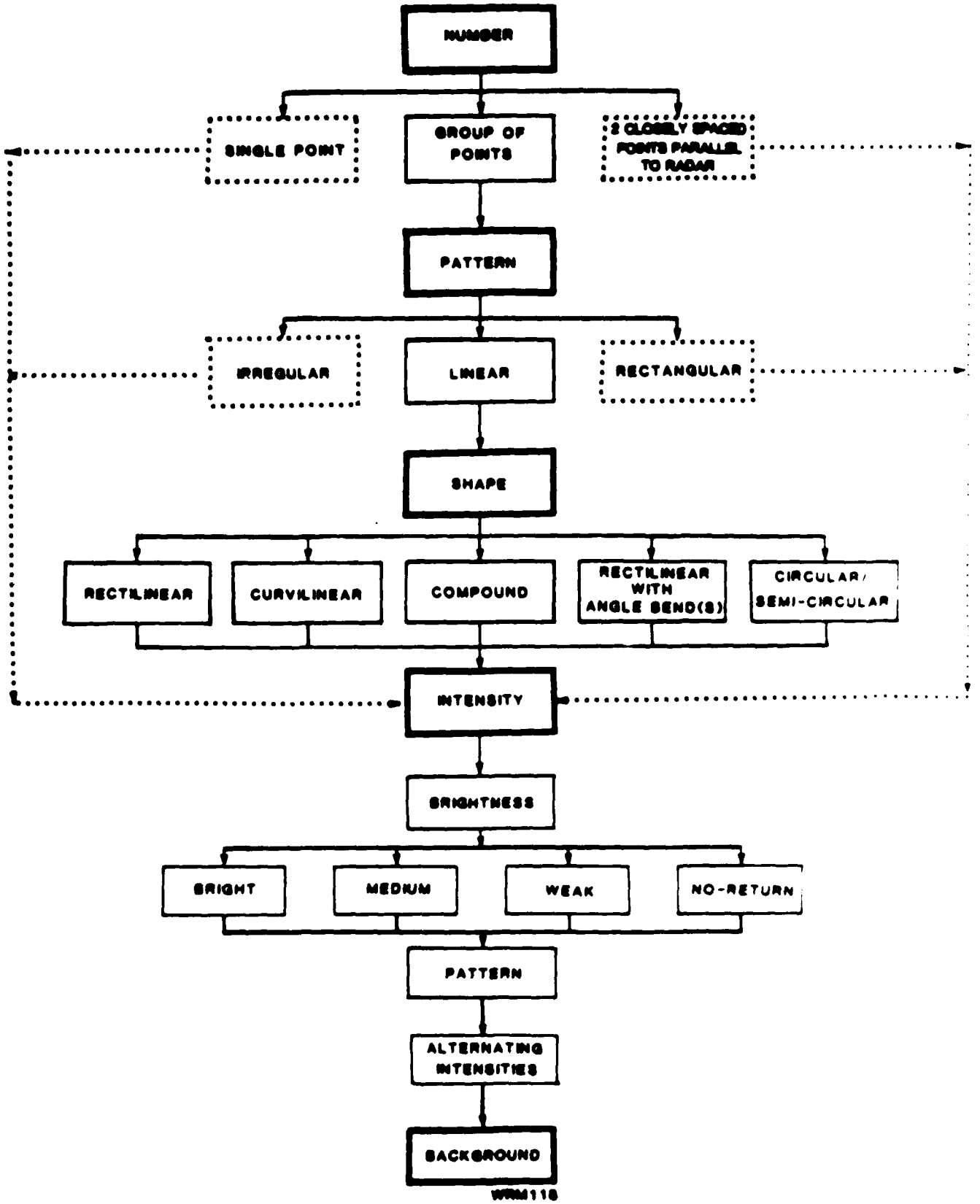


Figure 14. Flow Chart Used to Identify Point Descriptor Sets

The results of the computer vision study were examined in two categories: raster processing and vector processing. In this phase of the study, emphasis was placed on the results produced by the software, and little concern was given to the efficiency of the software. If the methodology is found to be workable, speed can always be increased by modifying the software and/or hardware used during analysis.

4.1 Raster Processing

Raster processing produced a thinned image containing feature edges. Currently, the algorithms are computationally intense, but if speed is a concern, they can be run at video rates using a feedback or array processor. It is expected that two areas of difficulty will be encountered after texture enhancement of the imagery. The first will lie in separating areas of high spatial variation from edges. This problem did not arise during the course of the investigation because texture could not be represented adequately by line-drawn graphics and so was not used. It will, however, be a consideration when analysis of radar imagery begins in the following phase. A second problem will be in determining the threshold that separates edges from the rest of the image during creation of the binary image. Texture-enhanced images of many of the descriptor examples were analyzed, and a method was found that successfully identified edges in the majority of examples. Difficulties occurred only when two primitives were very close and their edges merged after texture enhancement and thresholding.

4.2 Vector Processing

4.2.1 Point Features

All point features that were tested were successfully identified. However, point descriptors were found to contain patterns that were more complex than those encountered with line and area descriptors. Points were identified on the basis of the ratio of eigenvalues, generated from feature element coordinates, and on the basis of the number of feature elements per primitive. Brightness and background were easily calculated using the means and variances of the x,y coordinates of the feature pixels defining the edge of the point.

4.2.2 Line Feature

Line feature patterns were not as complex as points, but some of their characteristics were more difficult to identify. Line brightness and background were more difficult to determine because of the narrowness of many of the lines and the offset of the edges that occurs during thinning. Line orientation and parallelism were also new factors to be considered but were relatively easy to identify using eigenvalues and eigenvectors. Small, semi-circular lines were sometimes confused with rectilinear lines having angular bends. This was a result of the polygon approximation, which produced sharp angles if the error tolerances were too slack, but which misclassified lines with angular bends as curvilinear lines if the error tolerance was too tight. However, error tolerances were found that allowed correct identification of these line types although the separation between types was small. Compound lines were difficult to identify, but this will be corrected when the definition is clarified and made more rigorous.

4.2.3 Area Features

Area patterns were very similar to line patterns and were therefore easy to identify. Object and background brightness were determined in a manner similar to the way in which they were determined for points. The most difficult area characteristic to identify, and the most difficult descriptor in the whole study, was texture. This was due to two circumstances: 1) the texture of only a few graphic images could be digitized well. The fine textures were smoothed during digitization and appeared as a grey background; and 2) it was felt that the images that did show some texture were not good representations of the texture found in radar imagery. Therefore it was thought that the investigator's time would be better spent in areas that would prove more useful during the analysis of radar imagery in the next phase. Another characteristic of areas that had not been encountered in earlier studies was that of interlocking. This contiguity of areas having different brightness values and textures dramatically increased the computational complexity of the topological sorting, but it is not conceptually difficult to solve. Oval areas have not been separated at this time, but an algorithm has been developed which has not yet been incorporated into the system.

4.3 Summary

The heuristic approach that was taken enabled the investigator to reliably identify more than 90 percent of the descriptors provided. Of the 42 descriptors that were tested, all but four were classified correctly. Those not classified correctly were: the compound line, two images with different textures (one interlocking) and the oval area. As stated earlier, the compound line and oval area errors can be easily corrected. More work is needed in texture analysis, although a software foundation has been laid that can be enhanced in the next phase and used for the analysis of texture in radar imagery. The descriptors that were omitted from testing were those that strongly resembled some of the descriptors that were used (e.g., the graphic of the four parallel lines was tested, while, the graphic of the two parallel lines was not). In the next phase of the project, raster processing of the radar imagery will be the most crucial area of development in the computer-vision system, and image segmentation and texture analysis will be the most thoroughly reworked areas. Once a thinned image has been produced, the raster-to-vector conversion and vector processing should remain nearly intact.

REFERENCES

Pascucci, R.F., and E.T. Huffman, 1984, Development of Descriptor Sets for the Unambiguous Characterization of Geographic Features on SAR Imagery, U.S. Army Engineer Topographic Laboratories, Ft. Belvoir, VA, ETL-0369.

Pascucci, R.F., 1986, An Expert System for the Computer-Assisted Identification of Features on SAR Imagery, U.S. Army Engineer Topographic Laboratories, Ft. Belvoir, VA, ETL-0415.

Rhode, F.W., 1981, A Study of the Human Visual System in Support of Automated Feature Extraction, U.S. Army Engineer Topographic Laboratories, Ft. Belvoir, VA, AD-A109.

Pavlidis, T., 1982. Algorithms for Graphics and Image Processing, Computer Science Press, Maryland.

Speck, P.T., 1980. "Automated Recognition of Line Structures on Noisy Raster Images Applied to Electron Micrographs of DNA", IN: Proc. IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition, 604-609.

Williams, 1978. "An Efficient Algorithm for the Piecewise Linear of a Digitized Contour", Computer Graphics and Image Processing, 8: 286-293.

Williams, 1981. "Bounded Straight-Line Approximation of Digitized Planar Curves and Lines", Computer Graphics and Image Processing, 16: 370-381.

Woetzel, G., 1978. "A First and Economic Scan-to-Line Conversion Algorithm", Computer Graphics 12:3, 125-129.

APPENDIX A

TWIG, PRIMITIVE, AND SEGMENT FILE FORMATS

APPENDIX A

The following is a description of the operations of the software and outlines the ways in which subsections of an image can be analyzed by executing the various software modules as shown in the example below.

Ex. Given - 1) the image to be analyzed is being displayed in channel 0;
2) subsection 8 is to be analyzed;
3) the image on disk are stored in "../img".

```
TEXTURE    0        8   8
EDGE       0        8   8
THINT      0        8   8
THINT      0        8   8
THINT      0        8   8

COLLECT    8        lin8
FIXPRIM             lin8
NEW                 lin8
DECISION1           lin8
FIXSPLIT            lin8
FIXMERGE            lin8
FIXHOLE             lin8
FIXMERGE2           lin8
FIXAREA             lin8
FIXMERGE3           lin8
FIXAREA2            lin8
FIXHOLE             lin8
NEW2                lin8
FIXCRV              lin8
DECISION2           lin8
FIXSEG              lin8
FIXSEG              lin8
FIXPAT              lin8
FIXCLR              lin8 ../img/lines
FINAL               lin8
```

The user is not held to the parameters used in this example, but modifications to the example sequence must be consistent throughout. For example, the image may be in channel 0 or 1 but must be in one of the two, and any subsection can be analyzed from 0 in the upper-left corner to 15 in the lower-right corner.

Two command files have also been provided to shorten the command sequence. The following command sequence performs the same operation as the sequence shown in the previous example:

```
PROCESS  0  8  8
SEARCH   8  lin8 ../img/lines
```

File names commonly used are PTSxx, LINxx, and ARxx for point, line and area descriptors, respectively. The subsection being analyzed is inserted in the place of the "xx" shown above and ranges from 0 to 15. For example, lin8 was a file name used to analyze a line pattern in subsection 8. The following output files are created during execution of the above sequences:

Binary	Ascii
lin8.k00	lin8.asc
lin8.k01	lin8.prm
lin8.f01	lin8.dc1
lin8.k02	lin8.sp1
lin8.k03	lin8.mrg
lin8.k04	lin8.hle
lin8.k05	lin8.fin
lin8.k06	lin8.are
lin8.k07	lin8.las
lin8.k08	lin8.ar2
lin8.k09	lin8.hl2
lin8.kp1	lin8.sts
lin8.f03	lin8.crv
lin8.k10	lin8.dc2
lin8.f04	lin8.seg
lin8.k11	lin8.pat
lin8.k12	lin8.clr
lin8.k13	
lin8.k14	

Two files are created when any software module terminates. As discussed earlier, this creates a large number of files that must be deleted periodically. To save the results of any analysis, only the "*.k14" file must be saved. To display the results of the analysis to the screen, the user must type "final *" where "*" is the file name chosen earlier.

Figure A-1 shows the ascii results from executing "collect 8 lin8". Twenty-four twigs have been collected along with feature pixel coordinates and other pertinent data. Figure A-2 shows the results from grouping twigs into primitives and removing split and merge elements. Note that one primitive has been identified and the original 24 twigs have been consolidated to 15. Figure A-3 shows the primitive file after all topological sorting is complete; the 15 twigs have been reduced to one. Figure A-4 shows the final results from grouping primitives into segments. In order to save space, a segment was chosen that contained only one primitive. The header record information is shown at the beginning of the output file and contains information outlined in Figure A-5.

TABLE A-I.

Twig File Resulting from Raster-to-Vector Conversion

ntwig = 24

rake[curx] = 2
npts[rake[curx]] = 2
begtype[rake[curx]] = 2
endtype[rake[curx]] = 4
twig x, y = 96 35
twig x, y = 95 36

rake[curx] = 1
npts[rake[curx]] = 58
begtype[rake[curx]] = 1
endtype[rake[curx]] = 4
twig x, y = 38 35
twig x, y = 39 36
twig x, y = 40 36
twig x, y = 41 36
twig x, y = 42 36
twig x, y = 43 36
twig x, y = 44 36
twig x, y = 45 36
twig x, y = 46 36
twig x, y = 47 36
twig x, y = 48 36
twig x, y = 49 36
twig x, y = 50 36
twig x, y = 51 36
twig x, y = 52 36
twig x, y = 53 36
twig x, y = 54 36
twig x, y = 55 36
twig x, y = 56 36
twig x, y = 57 36
twig x, y = 58 36
twig x, y = 59 36
twig x, y = 60 36
twig x, y = 61 36
twig x, y = 62 36
twig x, y = 63 36
twig x, y = 64 36
twig x, y = 65 36
twig x, y = 66 36
twig x, y = 67 36
twig x, y = 68 36
twig x, y = 69 36
twig x, y = 70 36
twig x, y = 71 36
twig x, y = 72 36
twig x, y = 73 36
twig x, y = 74 36
twig x, y = 75 36
twig x, y = 76 36
twig x, y = 77 36
twig x, y = 78 36
twig x, y = 79 36
twig x, y = 80 36
twig x, y = 81 36
twig x, y = 82 36
twig x, y = 83 36
twig x, y = 84 36
twig x, y = 85 36
twig x, y = 86 36
twig x, y = 87 36
twig x, y = 88 36
twig x, y = 89 36
twig x, y = 90 36
twig x, y = 91 36
twig x, y = 92 36
twig x, y = 93 36
twig x, y = 94 36
twig x, y = 95 36

rake[curx] = 3
npts[rake[curx]] = 3
begtype[rake[curx]] = 2
endtype[rake[curx]] = 4
twig x, y = 96 35
twig x, y = 97 36
twig x, y = 96 37

rake[curx] = 4
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twig x, y = 95 36
twig x, y = 96 37

rake[curx] = 5
npts[rake[curx]] = 19
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twig x, y = 96 37
twig x, y = 96 38
twig x, y = 96 39
twig x, y = 96 40
twig x, y = 96 41
twig x, y = 96 42
twig x, y = 96 43
twig x, y = 96 44
twig x, y = 96 45
twig x, y = 96 46
twig x, y = 96 47
twig x, y = 96 48
twig x, y = 96 49
twig x, y = 96 50
twig x, y = 96 51
twig x, y = 96 52
twig x, y = 96 53
twig x, y = 96 54
twig x, y = 96 55

rake[curx] = 6
npts[rake[curx]] = 9
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twig x, y = 96 55
twig x, y = 95 56
twig x, y = 95 57
twig x, y = 95 58
twig x, y = 95 59
twig x, y = 95 60
twig x, y = 95 61
twig x, y = 95 62
twig x, y = 95 63

rake[curx] = 11
npts[rake[curx]] = 30
begtype[rake[curx]] = 2
endtype[rake[curx]] = 4
twig x, y = 66 63
twig x, y = 67 63
twig x, y = 68 63
twig x, y = 69 63
twig x, y = 70 63
twig x, y = 71 63
twig x, y = 72 63
twig x, y = 73 63
twig x, y = 74 63
twig x, y = 75 63
twig x, y = 76 63
twig x, y = 77 63
twig x, y = 78 63
twig x, y = 79 63
twig x, y = 80 63
twig x, y = 81 63
twig x, y = 82 63
twig x, y = 83 63
twig x, y = 84 63
twig x, y = 85 63
twig x, y = 86 63
twig x, y = 87 63
twig x, y = 88 63
twig x, y = 89 63
twig x, y = 90 63
twig x, y = 91 63
twig x, y = 92 63
twig x, y = 93 63
twig x, y = 94 63
twig x, y = 95 63

rake[curx] = 7
npts[rake[curx]] = 9
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twig x, y = 96 55
twig x, y = 97 56
twig x, y = 97 57
twig x, y = 97 58
twig x, y = 97 59
twig x, y = 97 60
twig x, y = 97 61
twig x, y = 97 62
twig x, y = 96 63

rake[curx] = 13
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twig x, y = 95 63
twig x, y = 96 63

rake[curx] = 8
npts[rake[curx]] = 2
begtype[rake[curx]] = 2
endtype[rake[curx]] = 4
twig x, y = 49 63
twig x, y = 48 64

rake[curx] = 17
npts[rake[curx]] = 4
begtype[rake[curx]] = 2
endtype[rake[curx]] = 4
twig x, y = 45 64
twig x, y = 46 64
twig x, y = 47 64
twig x, y = 48 64

rake[curx] = 9
npts[rake[curx]] = 15
begtype[rake[curx]] = 2

TABLE A-I. (cont'd)

endtype[rake[curx]] = 4
twig x, y = 49 63
twig x, y = 50 63
twig x, y = 51 63
twig x, y = 52 63
twig x, y = 53 63
twig x, y = 54 63
twig x, y = 55 63
twig x, y = 56 63
twig x, y = 57 63
twig x, y = 58 63
twig x, y = 59 63
twig x, y = 60 63
twig x, y = 61 63
twig x, y = 62 63
twig x, y = 63 64

rake[curx] = 10
npts[rake[curx]] = 2
begtype[rake[curx]] = 2
endtype[rake[curx]] = 4
twig x, y = 66 63
twig x, y = 65 64

rake[curx] = 20
npts[rake[curx]] = 3
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twig x, y = 63 64
twig x, y = 64 64
twig x, y = 65 64

rake[curx] = 12
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twig x, y = 95 63
twig x, y = 95 64

rake[curx] = 14
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twig x, y = 96 63
twig x, y = 96 64

rake[curx] = 23
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twig x, y = 95 64
twig x, y = 96 64

rake[curx] = 16
npts[rake[curx]] = 2
begtype[rake[curx]] = 2
endtype[rake[curx]] = 5
twig x, y = 45 64
twig x, y = 44 65

rake[curx] = 15
npts[rake[curx]] = 7
begtype[rake[curx]] = 1
endtype[rake[curx]] = 5
twig x, y = 38 64
twig x, y = 39 65
twig x, y = 40 65
twig x, y = 41 65
twig x, y = 42 65
twig x, y = 43 65
twig x, y = 44 65

rake[curx] = 19
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 5
twig x, y = 63 64
twig x, y = 62 65

rake[curx] = 18
npts[rake[curx]] = 15
begtype[rake[curx]] = 4
endtype[rake[curx]] = 5
twig x, y = 48 64
twig x, y = 49 65
twig x, y = 50 65
twig x, y = 51 65
twig x, y = 52 65
twig x, y = 53 65
twig x, y = 54 65
twig x, y = 55 65
twig x, y = 56 65
twig x, y = 57 65
twig x, y = 58 65
twig x, y = 59 65
twig x, y = 60 65
twig x, y = 61 65
twig x, y = 62 65

rake[curx] = 22
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 5
twig x, y = 95 64
twig x, y = 94 65

rake[curx] = 21
npts[rake[curx]] = 30
begtype[rake[curx]] = 4
endtype[rake[curx]] = 5
twig x, y = 65 64
twig x, y = 66 65
twig x, y = 67 65
twig x, y = 68 65
twig x, y = 69 65
twig x, y = 70 65
twig x, y = 71 65
twig x, y = 72 65
twig x, y = 73 65
twig x, y = 74 65
twig x, y = 75 65
twig x, y = 76 65
twig x, y = 77 65
twig x, y = 78 65
twig x, y = 79 65
twig x, y = 80 65
twig x, y = 81 65
twig x, y = 82 65
twig x, y = 83 65
twig x, y = 84 65
twig x, y = 85 65
twig x, y = 86 65
twig x, y = 87 65
twig x, y = 88 65
twig x, y = 89 65
twig x, y = 90 65
twig x, y = 91 65
twig x, y = 92 65
twig x, y = 93 65
twig x, y = 94 65

rake[curx] = 24
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 6
twig x, y = 96 64
twig x, y = 97 65

]]

TABLE A-II.

File Resulting from Grouping Twigs into Primitives and the Removal of Split and Merge Elements

nprim = 1

Primitive = 1

ntwig = 15

primtype = 0

rake[curx] = 1
npts[rake[curx]] = 58
begtype[rake[curx]] = 1
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 38 35
twig x, y = 39 36
twig x, y = 40 36
twig x, y = 41 36
twig x, y = 42 36
twig x, y = 43 36
twig x, y = 44 36
twig x, y = 45 36
twig x, y = 46 36
twig x, y = 47 36
twig x, y = 48 36
twig x, y = 49 36
twig x, y = 50 36
twig x, y = 51 36
twig x, y = 52 36
twig x, y = 53 36
twig x, y = 54 36
twig x, y = 55 36
twig x, y = 56 36
twig x, y = 57 36
twig x, y = 58 36
twig x, y = 59 36
twig x, y = 60 36
twig x, y = 61 36
twig x, y = 62 36
twig x, y = 63 36
twig x, y = 64 36
twig x, y = 65 36
twig x, y = 66 36
twig x, y = 67 36
twig x, y = 68 36
twig x, y = 69 36
twig x, y = 70 36
twig x, y = 71 36
twig x, y = 72 36
twig x, y = 73 36
twig x, y = 74 36
twig x, y = 75 36
twig x, y = 76 36
twig x, y = 77 36
twig x, y = 78 36
twig x, y = 79 36
twig x, y = 80 36
twig x, y = 81 36
twig x, y = 82 36
twig x, y = 83 36
twig x, y = 84 36
twig x, y = 85 36
twig x, y = 86 36
twig x, y = 87 36
twig x, y = 88 36
twig x, y = 89 36
twig x, y = 90 36
twig x, y = 91 36
twig x, y = 92 36
twig x, y = 93 36
twig x, y = 94 36
twig x, y = 95 36

rake[curx] = 2
npts[rake[curx]] = 4
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 95 36
twig x, y = 96 35
twig x, y = 97 36
twig x, y = 96 37

rake[curx] = 3
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 95 36
twig x, y = 96 37

rake[curx] = 4
npts[rake[curx]] = 19
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 96 37
twig x, y = 96 38
twig x, y = 96 39
twig x, y = 96 40
twig x, y = 96 41
twig x, y = 96 42
twig x, y = 96 43
twig x, y = 96 44
twig x, y = 96 45
twig x, y = 96 46
twig x, y = 96 47
twig x, y = 96 48
twig x, y = 96 49
twig x, y = 96 50
twig x, y = 96 51
twig x, y = 96 52
twig x, y = 96 53
twig x, y = 96 54
twig x, y = 96 55

rake[curx] = 5
npts[rake[curx]] = 9
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 96 55
twig x, y = 95 56
twig x, y = 95 57
twig x, y = 95 58
twig x, y = 95 59
twig x, y = 95 60
twig x, y = 95 61
twig x, y = 95 62
twig x, y = 95 63

rake[curx] = 6
npts[rake[curx]] = 9
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 96 55
twig x, y = 97 56
twig x, y = 97 57
twig x, y = 97 58
twig x, y = 97 59
twig x, y = 97 60
twig x, y = 97 61
twig x, y = 97 62
twig x, y = 96 63

rake[curx] = 7
npts[rake[curx]] = 31
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 95 63
twig x, y = 94 63
twig x, y = 93 63
twig x, y = 92 63
twig x, y = 91 63
twig x, y = 90 63
twig x, y = 89 63
twig x, y = 88 63
twig x, y = 87 63
twig x, y = 86 63
twig x, y = 85 63
twig x, y = 84 63
twig x, y = 83 63
twig x, y = 82 63
twig x, y = 81 63
twig x, y = 80 63
twig x, y = 79 63
twig x, y = 78 63
twig x, y = 77 63
twig x, y = 76 63
twig x, y = 75 63
twig x, y = 74 63
twig x, y = 73 63
twig x, y = 72 63
twig x, y = 71 63
twig x, y = 70 63
twig x, y = 69 63
twig x, y = 68 63
twig x, y = 67 63
twig x, y = 66 63
twig x, y = 65 64

rake[curx] = 8
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 95 63
twig x, y = 95 64

rake[curx] = 9
npts[rake[curx]] = 2
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 95 63
twig x, y = 96 63

rake[curx] = 10
npts[rake[curx]] = 3
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4

TABLE A-II. (cont'd)

twigtype[rake[curx]] = 0
twig x, y = 96 63
twig x, y = 96 64
twig x, y = 95 64

rake[curx] = 11
npts[rake[curx]] = 31
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 95 64
twig x, y = 94 65
twig x, y = 93 65
twig x, y = 92 65
twig x, y = 91 65
twig x, y = 90 65
twig x, y = 89 65
twig x, y = 88 65
twig x, y = 87 65
twig x, y = 86 65
twig x, y = 85 65
twig x, y = 84 65
twig x, y = 83 65
twig x, y = 82 65
twig x, y = 81 65
twig x, y = 80 65
twig x, y = 79 65
twig x, y = 78 65
twig x, y = 77 65
twig x, y = 76 65
twig x, y = 75 65
twig x, y = 74 65
twig x, y = 73 65
twig x, y = 72 65
twig x, y = 71 65
twig x, y = 70 65
twig x, y = 69 65
twig x, y = 68 65
twig x, y = 67 65
twig x, y = 66 65
twig x, y = 65 64

rake[curx] = 12
npts[rake[curx]] = 3
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 63 64
twig x, y = 64 64
twig x, y = 65 64

rake[curx] = 13
npts[rake[curx]] = 16
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 63 64
twig x, y = 62 63
twig x, y = 61 63
twig x, y = 60 63
twig x, y = 59 63
twig x, y = 58 63
twig x, y = 57 63
twig x, y = 56 63
twig x, y = 55 63
twig x, y = 54 63
twig x, y = 53 63
twig x, y = 52 63
twig x, y = 51 63
twig x, y = 50 63
twig x, y = 49 63
twig x, y = 48 64

rake[curx] = 14
npts[rake[curx]] = 16
begtype[rake[curx]] = 4
endtype[rake[curx]] = 4
twigtype[rake[curx]] = 0
twig x, y = 63 64
twig x, y = 62 65
twig x, y = 61 65
twig x, y = 60 65
twig x, y = 59 65
twig x, y = 58 65
twig x, y = 57 65
twig x, y = 56 65
twig x, y = 55 65
twig x, y = 54 65
twig x, y = 53 65
twig x, y = 52 65
twig x, y = 51 65
twig x, y = 50 65
twig x, y = 49 65
twig x, y = 48 64

rake[curx] = 15
npts[rake[curx]] = 11
begtype[rake[curx]] = 4
endtype[rake[curx]] = 1
twigtype[rake[curx]] = 0
twig x, y = 48 64
twig x, y = 47 64
twig x, y = 46 64
twig x, y = 45 64
twig x, y = 44 65
twig x, y = 43 65
twig x, y = 42 65
twig x, y = 41 65
twig x, y = 40 65
twig x, y = 39 65
twig x, y = 38 64
x]]

TABLE A-III.

Primitive File After Topological Sorting is Complete

nprim = 1

Primitive = 1

ntwig = 1

primtype = 0

twig: 1
npts[twignum] = 144
begtype[twignum] = 1
endtype[twignum] = 1
twigtype[twignum] = 1
twig x, y = 38 35
twig x, y = 39 36
twig x, y = 40 36
twig x, y = 41 36
twig x, y = 42 36
twig x, y = 43 36
twig x, y = 44 36
twig x, y = 45 36
twig x, y = 46 36
twig x, y = 47 36
twig x, y = 48 36
twig x, y = 49 36
twig x, y = 50 36
twig x, y = 51 36
twig x, y = 52 36
twig x, y = 53 36
twig x, y = 54 36
twig x, y = 55 36
twig x, y = 56 36
twig x, y = 57 36
twig x, y = 58 36
twig x, y = 59 36
twig x, y = 60 36
twig x, y = 61 36
twig x, y = 62 36
twig x, y = 63 36
twig x, y = 64 36
twig x, y = 65 36
twig x, y = 66 36
twig x, y = 67 36
twig x, y = 68 36
twig x, y = 69 36
twig x, y = 70 36
twig x, y = 71 36
twig x, y = 72 36
twig x, y = 73 36
twig x, y = 74 36
twig x, y = 75 36
twig x, y = 76 36
twig x, y = 77 36
twig x, y = 78 36
twig x, y = 79 36
twig x, y = 80 36
twig x, y = 81 36
twig x, y = 82 36
twig x, y = 83 36
twig x, y = 84 36
twig x, y = 85 36
twig x, y = 86 36
twig x, y = 87 36
twig x, y = 88 36
twig x, y = 89 36
twig x, y = 90 36
twig x, y = 91 36
twig x, y = 92 36
twig x, y = 93 36
twig x, y = 94 36
twig x, y = 95 36
twig x, y = 96 37
twig x, y = 96 38
twig x, y = 96 39
twig x, y = 96 40
twig x, y = 96 41
twig x, y = 96 42
twig x, y = 96 43
twig x, y = 96 44
twig x, y = 96 45
twig x, y = 96 46
twig x, y = 96 47
twig x, y = 96 48
twig x, y = 96 49
twig x, y = 96 50
twig x, y = 96 51
twig x, y = 96 52
twig x, y = 96 53
twig x, y = 96 54
twig x, y = 96 55
twig x, y = 96 56
twig x, y = 96 57
twig x, y = 96 58
twig x, y = 95 59
twig x, y = 95 60
twig x, y = 95 61
twig x, y = 95 62
twig x, y = 95 63
twig x, y = 96 64
twig x, y = 95 64
twig x, y = 94 64
twig x, y = 93 64
twig x, y = 92 64
twig x, y = 91 64
twig x, y = 90 64
twig x, y = 89 64
twig x, y = 88 64
twig x, y = 87 64
twig x, y = 86 64
twig x, y = 85 64
twig x, y = 84 64
twig x, y = 83 64
twig x, y = 82 64
twig x, y = 81 64
twig x, y = 80 64
twig x, y = 79 64
twig x, y = 78 64
twig x, y = 77 64
twig x, y = 76 64
twig x, y = 75 64
twig x, y = 74 64
twig x, y = 73 64
twig x, y = 72 64
twig x, y = 71 64
twig x, y = 70 64
twig x, y = 69 64
twig x, y = 68 64
twig x, y = 67 64
twig x, y = 66 64
twig x, y = 65 64
twig x, y = 64 64
twig x, y = 63 64
twig x, y = 62 64
twig x, y = 61 64
twig x, y = 60 64
twig x, y = 59 64
twig x, y = 58 64
twig x, y = 57 64
twig x, y = 56 64
twig x, y = 55 64
twig x, y = 54 64
twig x, y = 53 64
twig x, y = 52 64
twig x, y = 51 64
twig x, y = 50 64
twig x, y = 49 64
twig x, y = 48 64
twig x, y = 47 64
twig x, y = 46 64
twig x, y = 45 64
twig x, y = 44 65
twig x, y = 43 65
twig x, y = 42 65
twig x, y = 41 65
twig x, y = 40 65
twig x, y = 39 65
twig x, y = 38 64

TABLE A-IV.

Segment File Resulting from Analysis of Primitive File

Segment type = 2

Number of primitives = 1

Primitive shape = 3

Segment pattern = 0

Pattern shape = 0

Orientation = -1

Dimension = 0

Intensity = 3

Texture = 0

Background = 5

Primitive: 1
****
Number of twigs: 1
Primtype: 3

Twig number: 1
npts[j] = 144
begtype[j] = 1
endtype[j] = 1
twigtype[j] = 1
twig x, y = 38 35
twig x, y = 39 36
twig x, y = 40 36
twig x, y = 41 36
twig x, y = 42 36
twig x, y = 43 36
twig x, y = 44 36
twig x, y = 45 36
twig x, y = 46 36
twig x, y = 47 36
twig x, y = 48 36
twig x, y = 49 36
twig x, y = 50 36
twig x, y = 51 36
twig x, y = 52 36
twig x, y = 53 36
twig x, y = 54 36
twig x, y = 55 36
twig x, y = 56 36
twig x, y = 57 36
twig x, y = 58 36
twig x, y = 59 36
twig x, y = 60 36
twig x, y = 61 36
twig x, y = 62 36
twig x, y = 63 36
twig x, y = 64 36
twig x, y = 65 36
twig x, y = 66 36
twig x, y = 67 36
twig x, y = 68 36
twig x, y = 69 36
twig x, y = 70 36
twig x, y = 71 36
twig x, y = 72 36
twig x, y = 73 36
twig x, y = 74 36
twig x, y = 75 36
twig x, y = 76 36
twig x, y = 77 36
twig x, y = 78 36
twig x, y = 79 36
twig x, y = 80 36
twig x, y = 81 36
twig x, y = 82 36
twig x, y = 83 36
twig x, y = 84 36
twig x, y = 85 36
twig x, y = 86 36
twig x, y = 87 36
twig x, y = 88 36
twig x, y = 89 36
twig x, y = 90 36
twig x, y = 91 36
twig x, y = 92 36
twig x, y = 93 36
twig x, y = 94 36
twig x, y = 95 36
twig x, y = 96 37
twig x, y = 96 38
twig x, y = 96 39
twig x, y = 96 40
twig x, y = 96 41
twig x, y = 96 42
twig x, y = 96 43
twig x, y = 96 44
twig x, y = 96 45
twig x, y = 96 46
twig x, y = 96 47
twig x, y = 96 48
twig x, y = 96 49
twig x, y = 96 50
twig x, y = 96 51
twig x, y = 96 52
twig x, y = 96 53
twig x, y = 96 54
twig x, y = 96 55
twig x, y = 96 56
twig x, y = 96 57
twig x, y = 96 58
twig x, y = 95 59
twig x, y = 95 60
twig x, y = 95 61
twig x, y = 95 62
twig x, y = 95 63
twig x, y = 96 64
twig x, y = 95 64
twig x, y = 94 64
twig x, y = 93 64
twig x, y = 92 64
twig x, y = 91 64
twig x, y = 90 64
twig x, y = 89 64
twig x, y = 88 64
twig x, y = 87 64
twig x, y = 86 64
twig x, y = 85 64
twig x, y = 84 64
twig x, y = 83 64
twig x, y = 82 64
twig x, y = 81 64
twig x, y = 80 64
twig x, y = 79 64
twig x, y = 78 64
twig x, y = 77 64
twig x, y = 76 64
twig x, y = 75 64
twig x, y = 74 64
twig x, y = 73 64
twig x, y = 72 64
twig x, y = 71 64
twig x, y = 70 64
twig x, y = 69 64
twig x, y = 68 64
twig x, y = 67 64
twig x, y = 66 64
twig x, y = 65 64
twig x, y = 64 64
twig x, y = 63 64
twig x, y = 62 64
twig x, y = 61 64
twig x, y = 60 64
twig x, y = 59 64
twig x, y = 58 64
twig x, y = 57 64
twig x, y = 56 64
twig x, y = 55 64
twig x, y = 54 64
twig x, y = 53 64
twig x, y = 52 64
twig x, y = 51 64
twig x, y = 50 64
twig x, y = 49 64
twig x, y = 48 64
twig x, y = 47 64
twig x, y = 46 64
twig x, y = 45 64
twig x, y = 44 65
twig x, y = 43 65
twig x, y = 42 65
twig x, y = 41 65
twig x, y = 40 65
twig x, y = 39 65
twig x, y = 38 64

PROGRAM USED TO ID. DESCRIPTOR	DESCRIPTOR	VALUES	
FIXSEG	TYPE OF SEGMENT	POINT	YES, NO
		LINE	YES, NO
		AREA	YES, NO
	# OF PRIMITIVES	P	ANY NUMBER
		L	ANY NUMBER
		A	ANY NUMBER
	PRIMITIVE SHAPE	P	POINT
		L	STRAIGHT, ANGULAR, CURVILINEAR, SEMI/CIRC. OR CIRCULAR, COMPOUND & MIXED
		A	RECTANGULAR, POLYGONAL, OVAL/CIRCULAR, IRREGULAR, MIXED
FIXPAT	SEGMENT PATTERN	P	RECTANGULAR, LINEAR, IRREGULAR
		L	RECTANGULAR, LINEAR, IRREGULAR
		A	RECTANGULAR, LINEAR, INTERLOCKING
	PATTERN SHAPE	P	IF SEG. PATTERN IS LINEAR: STRAIGHT, ANGULAR, CURVILINEAR, SEMI/CIRC. OR CIRCULAR, COMPOUND
		L	N/A
		A	N/A
	ORIENTATION	P	N/A
		L	NO RELATIONSHIP TO RADAR, PARALLEL, CONVEX OR CONCAVE TO RADAR
		A	N/A
	DIMENSION	P	N/A
		L	LENGTH, WIDTH, SPACING: ANY NUMBER
		A	LENGTH, WIDTH, SPACING, DIAMETER: ANY NUMBER
FIXCLR	INTENSITY	P	DARK, DARK-MEDIUM, MEDIUM, MEDIUM-BRIGHT, BRIGHT
		L	DARK, DARK-MEDIUM, MEDIUM, MEDIUM-BRIGHT, BRIGHT
		A	DARK, DARK-MEDIUM, MEDIUM, MEDIUM-BRIGHT, BRIGHT
	TEXTURE	P	FINE, MEDIUM, COARSE
		L	FINE, MEDIUM, COARSE
		A	FINE, MEDIUM, COARSE
	BACKGROUND	P	DARK, DARK-MEDIUM, MEDIUM, MEDIUM-BRIGHT, BRIGHT
		L	DARK, DARK-MEDIUM, MEDIUM, MEDIUM-BRIGHT, BRIGHT
		A	DARK, DARK-MEDIUM, MEDIUM, MEDIUM-BRIGHT, BRIGHT

WRM123

Figure A-1. Information Contained Within Header Record of Segment File.

APPENDIX B

SOFTWARE DESCRIPTION

APPENDIX B

The following are descriptions of the software that was developed during the investigation:

Software Utilities

Author: Daniel Gordon
Autometric, Inc.

Program: c0

Description: Clears channel 0.

Input Parameters: None.

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: c1

Description: Clears channel 1.

Input Parameters: None.

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: clr

Description: Clears a 128 x 128 pixel subsection of channel 0 or 1.

Input Parameters: frame (channel) - 0 or 1, subsection - 0:15

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: color

Description: Sets a 128 x 128 pixel subsection to a particular brightness.

Input Parameters: subsection - 0:15, brightness - 0:255

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: display

Description: Displays an image or subsection of an image to the screen.

Input Parameters: filename - file to be displayed

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: gl

Description: Clears graphics from display screen.

Input Parameters: None.

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: gon

Description: Turns graphic planes on, overlaying channel 0 or 1.

Input Parameters: frame (channel) - 0 or 1

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: histo

Description: Generates a histogram for a 128 x 128 pixel area and stores it on disk.

Input Parameters: subsection - 0:15, filename

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Modification of program written by ETL

Program: histreg

Description: Generates a histogram of an area defined by a box cursor and displays the histogram on the screen.

Input Parameters: None.

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: lut

Description: Contrast stretching routine.

Input Parameters: frame (channel) - 0 or 1

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: m0

Description: Views channel 0.

Input Parameters: None.

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: m1

Description: Views channel 1.

Input Parameters: None.

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: save

Description: Saves an image being displayed to disk.

Input Parameters: disk filename

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: thresh

Description: Thresholds the input subsection using the threshold provided and writes it to output subsection creating a new binary image.

Input Parameters: input subsection - 0:15, output subsection - 0:15, threshold value - 0:255

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: zmsc

Description: Zoom and scroll routine.

Input Parameters: frame (channel) - 0 or 1

Required Libraries: radar.a, imagelib.a,sublib.a

Computer-Vision Software

Author: Daniel Gordon
Autometric, Inc.

Program: collect

Description: Transforms raster edges into vectors.

Input Parameters: input subsections, 0:15 filename

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: decision1

Description: Identifies points and straight lines in vector data.

Input Parameters: twig filename

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: decision2

Description: Distinguishes between lines with angular bends and curvilinear lines.

Input Parameters: twig filename

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: drawpoly

Description: Reads polygon data resulting from polygon approximation and writes vectors to graphics memory for visual analysis.

Input Parameters: frame - 0 or 1, output subsections 0:15, twig filename

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: drawprim

Description: Reads primitive file and writes primitive to graphic memory for visual analysis.

Input Parameters: frame - 0 or 1, output subsections 0:15, twig filename.

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: edge

Description: Thresholds an image, selecting the threshold automatically based on the image histogram; the threshold identifies areas of high spatial variation.

Input Parameters: frame (channel) - 0 or 1, input subsections 0:15, output subsection 0:15

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: final

Description: Converts information stored in the segment file header record into English language and displays on the terminal.

Input Parameters: twig framename

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: fixarea

Description: Topological sorting: joins together joints into curls.

Input Parameters: twig filename.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: fixclr

Description: Determines object brightness, texture and background brightness.

Input Parameters: twig filename, disk filename, input subsections 0:15.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: fixcrv

Description: Polygon approximation routine.

Input Parameters: twig filename.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: fixhole

Description: Removes small narrow curls from vector data.

Input Parameters: twig filename.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: fixmerge

Description: Removes merge elements from vector data.

Input Parameters: twig filename.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: fixpat

Description: Determines primitive patterns within a segment.

Input Parameters: twig filename.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: fixprim

Description: Reads twig file created by collect and groups twigs into primitives.

Input Parameters: twig filename.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: fixseg

Description: Groups primitives into segments.

Input Parameters: twig filename.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: fixsplit

Description: Topo sorting: removes split elements from vector data.

Input Parameters: twig filename.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: new

Description: Generates primitive statistics.

Input Parameters: None.

Required Libraries: libssp.a

Author: Daniel Gordon
Autometric, Inc.

Program: readprim

Description: Reads primitive files and writes contents to CRT.

Input Parameters: twig filename.

Required Libraries: None.

Author: Daniel Gordon
Autometric, Inc.

Program: readtwig

Description: Reads twig files and writes contents to CRT.

Input Parameters: twig filename.

Required Libraries: radar.a, imagelib.a,sublib.a

Author: Daniel Gordon
Autometric, Inc.

Program: texture

Description: Enhances texture of 128 x 128 pixel subsections of images by summing the absolute value of the difference between the center pixel in a 3 x 3 window and each of its eight surrounding neighbors.

Input Parameters: frame (channel) 0 - 1, input subsections 0:15, output subsection 0:15.

Required Libraries: radar.a, imagelib.a,sublib.a

END

9-87

DTIC